PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : G06F 17/60 | A2

(11) International Publication Number: WO 00/52618

(43) International Publication Date: 8 September 2000 (08.09.00)

(21) International Application Number: PCT/US00/05080

(22) International Filing Date: 29 February 2000 (29.02.00)

(30) Priority Data:
09/260,079 2 March 1999 (02.03.99) US

(71) Applicant: AURIGIN SYSTEMS, INC. [US/US]; 10710 North Tantau Avenue, Cupertino, CA 95014-0717 (US).

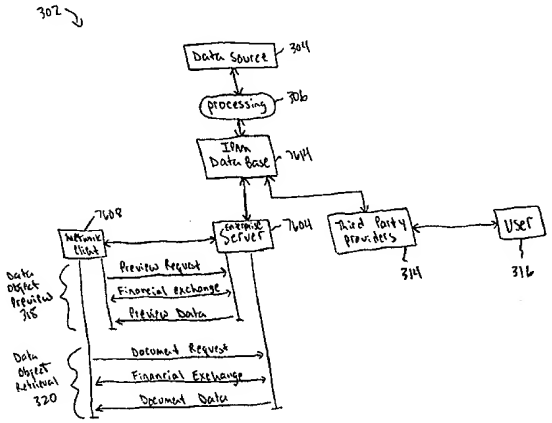
(72) Inventors: RIVETTE, Kevin, G.; 2165 Waverley Street, Palo Alto, CA 94303 (US). RAPPAPORT, Irving, S.; 1500 Edgewood Drive, Palo Alto, CA 94303 (US). HOHMANN, Luke; 306 Windmill Park Lane, Mountain View, CA 94043 (US). PUGLIA, David; 17429 East Vineland Avenue, Los Gatos, CA 95030 (US). DEWOLFE, Andrew, S.; 242 Acalanes Drive #11, Sunnyvale, CA 94086 (US). GORETSKY, David; 272 Waverly Street, Sunnyvale, CA 94086 (US). JACKSON, Adam; 1063 Morse Avenue #7-107, Sunnyvale, CA 94089 (US). KUROWSKI, Scott; 1038 Corvette Drive, San Jose, CA 95129 (US). PARK, Brian; 2636 Ponce Avenue, Belmont, CA 94002 (US). RABB, Charles, Jr.; 730 East Evelyn #638, Sunnyvale, CA 94086 (US). ROSENQUIST, Brent; 1668 Kennard Way, Sunnyvale, CA 94087 (US). SCHNITZ, Matthew; 2558 Mardell Way, Mountain View, CA 94043 (US). SMITH, David, W.; 3 Morning Sun Court, Mountain View, CA 94043 (US). PARADAN, Thierry; 1058 Paintbrush Drive, Sunnyvale, CA 94086 (US). BASHSHUR, Noura; 306 Windmill Park Lane, Mountain View, CA 94043 (US).

(74) Agents: LEE, Michael, Q. et al.; Sterne, Kessler, Goldstein & Fox P.L.L.C., Suite 600, 1100 New York Avenue, N.W., Washington, DC 20005-3934 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: INTELLECTUAL PROPERTY ASSET MANAGER (IPAM) FOR CONTEXT PROCESSING OF DATA OBJECTS



(57) Abstract

Context data processing is described herein. One or more contexts are selected. Each context includes one or more attributes, and a plurality of data objects that satisfy the attributes. A list of data objects contained in the selected contexts is displayed. At least some of the data objects in the selected contexts are processed. Such processing may involve generating claim trees, citation trees, and data object families, which may be displayed using hyperbolic trees. In an embodiment, the contexts are groups. In other embodiment, the contexts are each associated with a data object type. In this latter embodiment, the contexts include data objects of their respective data object types.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

Intellectual Property Asset Manager (IPAM) for Context Processing of Data Objects

Background of the Invention

Field of the Invention

The present application is a continuation-in-part of pending application "System, Method, and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, filed August 21, 1998, which is a continuation-in-part of pending application "Using Hyperbolic Trees to Visualize Data Generated By Patent-Centric and Group-Oriented Data Processing," Ser. No. 07/921,369, filed August 29, 1997, which is a continuation-in-part of allowed application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 07/867,392, filed June 2, 1997, all of which are incorporated herein by reference in their entireties.

The present invention is generally related to data processing, and more particularly related to, *inter alia*, processing data objects in a variety of contexts.

Related Art

Patents are becoming more and more important to a business's success, especially in today's global economy. Patents can be viewed as a new type of currency in this global economy because they grant the holder with a right to exclude others from making, using, or selling the patented technology. In some industries, product turnover is fairly rapid. However, core technology, product features, and markets change at a much slower rate. Accordingly, even in fast-moving industries, patents which cover core technology are very valuable at protecting a company's research and development investment for an extended period of time.

Patents are also valuable as revenue generators. In 1993, for example, the revenue generated from patents by U.S. companies was over $60 billion. Fred Warshofsky, *The Patent Wars*, John Wiley & Sons, Inc., New York, 1994. These patent revenue dollars are rising each year.

Patents are further valuable because they collectively represent a vast technological database. According to Larry Kahaner's book, *Competitive Intelligence*, Simon & Schuster, 1996, "More than 75 percent of the information contained in U.S. patents is never released anywhere else."

More and more corporations are recognizing the value of patents. The number of patents applied for and issued to U.S. companies is increasing every year, especially in fast moving industries such as computer software, telecommunications, and biotechnology. Many international companies have also recognized the value of patents. In fact, foreign companies regularly rank among the leaders in issued U.S. patents.

Yet, for all the heightened awareness being paid to patents in some quarters, patents remain one of the most underutilized assets in a company's portfolio. This is due, at least in significant part, to the fact that patent analysis, whether for purposes of licensing, infringement, enforcement, freedom to operate, technical research, product development, etc., is a very difficult, tedious, time consuming, and expensive task, particularly when performed with paper copies of patents. Often times, it is a difficult or even impossible task to simply identify relevant patents. Accordingly, detailed patent related analysis is usually not done, or it is done in an ad hoc, unorganized, incomplete, inefficient, and/or ineffective manner.

It would be very beneficial to have automated tools that aid users in processing and analyzing patent-related information and non-patent related information for making corporate business decisions.

Summary of the Invention

Briefly stated, an embodiment of the invention is directed to systems, methods, computer program products, and combinations thereof, of context data processing. The embodiment operates by selecting one or more contexts. Each context includes one or more attributes, and a plurality of data objects that satisfy the attributes. A list of data objects contained in the selected contexts is displayed. At least some of the data objects in the selected contexts are processed. Such processing may involve generating hierarchical and or directed acyclic graph data structures to represent relationships among the data objects. These data structures can then be displayed in a variety of well-known techniques including but not limited to hyperbolic trees. Examples of such hierarchical or directed acyclic graph structures include claim trees, citation trees, and data object families, which may be displayed using hyperbolic trees.

In an embodiment, the contexts are groups. In other embodiment, the contexts are each associated with a data object type. In this latter embodiment, the contexts include data objects of their respective data object types. The invention is not limited to these embodiments.

The invention supports the generation of annotations. The invention supports a plurality of annotation types, including document annotations, group annotations, data object type annotations, case annotations, and enterprise annotations. The invention also supports form-based annotations.

In an embodiment, the invention is directed to a system having an intellectual property asset manager (IPAM) and a plug-in manager coupled to the IPAM. The system also includes at least one plug-in coupled to the plug-in manager, and at least one external data processing component coupled to the plug-in. In an embodiment, the external data processing component displays data using at least graphs. In another embodiment, the external data processing

component displays data using at least maps. The plug-in manager has a first application programming interface (API), and the external data processing component each has a second API. The plug-in translates messages from the plug-in manager to the external data processing component to a format conforming to the second API, and translates messages from the external data processing component to the plug-in manager to a format conforming to the first API.

The invention is also directed to assignee name processing. Such processing operates by selecting a normalized assignee name for an entity, identifying name representations of the entity in a data set, and linking the name representations to the normalized assignee name.

Further features and advantages of the invention, as well as various embodiments of the invention, are described in detail below with reference to the accompanying drawings.

Brief Description of the Figures

The present invention will be described with reference to the accompanying drawings, wherein:

FIG. 1 illustrates an example group browser;

FIG. 2 illustrates a flowchart of image skimming according to an embodiment of the invention;

FIG. 3 depicts an event trace corresponding to a preview data object function according to an embodiment of the invention;

FIG. 4 illustrates a patent being a member of a plurality of groups;

FIG. 5 illustrates a data object, such as a document, being a member of a plurality of contexts;

FIG. 6 illustrates a corporate document window according to an embodiment of the invention;

FIG. 7 illustrates filtering in a corporate document window according to document kind;

FIG. 8 illustrates filtering in a corporate document window according to security class;

FIG. 9 illustrates a new corporate document kind dialog according to an embodiment of the invention;

FIGS. 10 and 11 illustrate a new security class dialog according to an embodiment of the invention;

FIG. 12 illustrates a new corporate document dialog according to an embodiment of the invention;

FIGS. 13-15 illustrate the addition of a corporate document to the IPAM system according to an embodiment of the invention;

FIG. 16 illustrates a context browser according to an embodiment of the invention;

FIG. 17 illustrates a type table according to an embodiment of the invention;

FIG. 18 illustrates a type document xref table according to an embodiment of the invention;

FIG. 19 illustrates an example type table;

FIG. 20 illustrates an example type document xref table;

FIGS. 21 and 22 are used to describe a data object preview function according to an embodiment of the invention;

FIG. 23 is a table illustrating the differences between different annotation types;

FIGS. 24-30 illustrate various types of document annotations according to embodiments of the invention;

FIG. 31 illustrates an annotation pane according to an embodiment of the invention;

FIG. 32 illustrates a flowchart representing the operation of the invention when creating an annotation that is linked to a data object;

FIG. 33 illustrates a flowchart representing the operation of the invention when creating a group or type annotation;

FIGS. 34-36 illustrate search GUIs according to embodiments of the invention;

FIG. 37 illustrates an example claim tree;

FIG. 38 illustrates a flowchart reflecting the operation of the invention when generating and utilizing a claim tree;

FIGS. 39-41 are used to describe a display claim branch function of the present invention;

FIG. 42A depicts a menu applicable to claim trees;

FIG. 42B illustrates node state information that is stored in nodes of hyperbolic trees according to an embodiment of the invention;

FIG. 43 illustrates an example patent citation tree;

FIG. 44 depicts a patent ref table according to an embodiment of the invention;

FIG. 45 illustrates a non-patent ref table according to an embodiment of the invention;

FIG. 46 illustrates an example patent ref table;

FIG. 47 illustrates an example non-patent ref table;

FIG. 48 illustrates a data object family table according to an embodiment of the invention;

FIG. 49 illustrates an example data object family table;

FIGS. 50A and 50B depict a table related to relationship types supported by embodiments of the present invention;

FIG. 51 illustrates an example patent family chronology;

FIG. 52 illustrates an example data object family table corresponding to the patent family chronology of FIG. 51;

FIG. 53 illustrates an example assignee technology patent family;

FIG. 54 illustrates an example data object family table corresponding to the assignee technology patent family of FIG. 53;

FIG. 55 illustrates a flowchart representing the operation of embodiments of the invention when generating relationship tables;

FIG. 56 is used to describe example sources of relationship data utilized by embodiments of the invention;

FIG. 57A is a flowchart representing the operation of the invention when analyzing relationship information;

FIG. 57B is a flowchart representing the operation of the invention when computing closure of a data object family;

FIG. 58 is a diagram illustrating the operation of the enterprise server when interacting with third party components and tools;

FIG. 59 is a flowchart illustrating the operation of the enterprise server when interacting with third party components/tools;

FIG. 60 is a more detailed block diagram of the enterprise server interacting with third party components/tools;

FIG. 61 is an event trace representing the operation of the enterprise server when interacting with third party components;

FIGS. 62A and 62B are a flowchart corresponding to the event trace diagram of FIG. 61;

FIG. 63 is an example data flow diagram relating to the operation of the enterprise server when interacting with a third party component that represents data using landscapes or maps;

FIG. 64 is a flowchart representing an example operation of a third party component interacting with the enterprise server;

FIGS. 65 and 66 represent example displays of data represented as maps or landscapes;

FIG. 67 is an example display of data represented using a graph with dynamically adjustable axis;

FIG. 68 is an event trace of the enterprise server operating with a specific third party component/tool;

FIGS. 69-73 depict various configurations and distributions of components and functionality supported by the present invention;

FIG. 74 is a flowchart representing the operation of embodiments of the invention when performing assignee name processing;

FIG. 75 illustrates another context browser according to embodiments of the invention;

FIG. 76 is a block diagram of the enterprise server according to embodiments of the invention;

FIG. 77 is a flowchart representing the operation of the invention when performing context processing according to embodiments of the invention.

FIGS. 78A and 78B illustrate free text annotations according to an embodiment of the invention;

FIG. 79 illustrates an example form-based annotation;

FIG. 80 illustrates another example formed based annotation;

FIG. 81 is a data flow diagram used to describe form-based annotations according to embodiments of the invention;

FIGS. 82, 83A, and 83B are flow charts representing the operation of form-based annotations according to embodiments of the invention;

FIG. 84 is a flow chart representing the operation of the invention when inputting an arbitrary data object;

FIG. 85 is a data flow diagram supporting the flow chart of FIG. 84;

FIG. 86 is a block diagram of an example computer system useful for implementing modules of the invention;

FIG. 87 illustrates an example patent claim tree module;

FIG. 88 illustrates an example landscape or map tool operable with embodiments of the invention;

FIG. 89 illustrates an example search tool operable with embodiments of the invention;

FIG. 90 illustrates an example export operation; and

FIGS. 91-98 are used to illustrate an example plug-in operation of the invention with a third party tool.

In the following text, reference is sometimes made to existing patents. Also, some of the figures reference or illustrate existing patents. For illustrative purposes, information from and/or about these patents has sometimes been modified or created in order to support the particular examples being discussed. Accordingly, the information provided herein about these existing patents should be considered to be fictional unless verified through comparison with copies of the actual patents that are available from national patent offices.

Detailed Description of the Preferred Embodiments

Overview of the Invention

The present invention is directed to systems, methods, computer program products, and components and combinations thereof, for processing data objects. Such processing includes, but is not limited to, acquiring data objects, organizing data objects, storing data objects, visualizing data objects, displaying data objects, manipulating data objects, annotating data objects, processing data contained in data objects, etc. The invention is also directed to combinations of the modules and functions described herein, and also described in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, herein incorporated by reference in their entireties.

In embodiments of the invention, the data objects include, but are not limited to, patents and patent-related data objects. More generally, the data objects include, but are not limited to, intellectual property related data objects. The data objects also include, but are not limited to, data objects of past, present, or potentially future interest to corporate entities. For illustrative purposes, the invention is sometimes described in terms of these embodiments. However, the invention is not limited to these embodiments. The invention is applicable to data objects of any type and subject matter. Accordingly, description of the invention in terms of these embodiments is provided for illustrative purposes only, and is not limiting.

The invention may be used with immutable data objects (i.e., data objects that cannot be modified), such as but not limited to data objects produced by government agencies and similar institutions. For example, an issued patent is generally considered an immutable data object since it is preferred that it not be modified. An executed contract is another example of a document that can be immutable. However, the invention is not limited to these embodiments. The invention can operate with mutable data objects, and/or data objects produced by private entities.

In embodiments of the invention, the data objects include documents, such as textual documents, image documents, and combinations thereof. For illustrative purposes, the invention is sometimes described in terms of these embodiments. However, the invention is not limited to these embodiments. The invention is applicable to data objects of any type and form, including but not limited to video, audio, sensory, multimedia, tactile, computer programs, links, etc., and combinations thereof. Accordingly, description of the invention in terms of these embodiments is provided for illustrative purposes only, and is not limiting.

In some embodiments, the present invention is intended to aid a corporate entity with planning over its entire enterprise. Accordingly, the present invention is sometimes referred to as an enterprise system and method (or an enterprise server). In some embodiments, the present invention is intended to aid a corporate entity in developing business-related strategies, plans, and actions. Accordingly, the present invention is sometimes referred to as a business decision system (BDS). In some embodiments, the present invention is intended to aid a corporate entity with managing intellectual property assets. Accordingly, the present invention is sometimes referred to as an intellectual property asset manager (IPAM). In any of these embodiments, the invention can run in a single user or a multi-user mode of operation.

Image Skimming

The invention supports an image skimming function. In accordance with the image skimming function, preferably, the first image page of each document in a collection of documents is displayed in succession pursuant to appropriate user command.

The invention is not limited to this embodiment. In particular, the skimming feature of the invention is not limited to document images. The skimming feature enables one to view or otherwise obtain information about a data object in a form that is consistent with that data object. For example, and without limitation, if a group contained a set of video clips, the skimming pane might allow for a video clip to be displayed/played directly from that window.

FIG. 1 illustrates an example user interface 102 according to embodiments of the invention. The user interface 102 is called a console or a browser.

A group hierarchy is shown in a group pane 104 of the browser 102. A group is a data structure that includes a collection of data objects, such as documents. The data objects in a group typically follow a common theme or characteristic. For example, a first group may include patents and other documents that map to a product being manufactured and sold by a company. A second group may include patents and other documents that map to a product being considered for future manufacture and sale by a company. A third group may include patents and other documents related to a research product. A group may contain other groups, or may be included in one or more groups. Accordingly, in embodiments of the invention, groups are hierarchically organized. Groups are further described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, herein incorporated by reference in its entirety.

As noted above, a group hierarchy is shown in a group pane 104 of the browser 102. In the example of FIG. 1, a group "771 docs" is selected, as

indicated as 105. A list of data objects contained in the selected group 105 is displayed in a contents pane 106. In the example of FIG. 1, a document "EP 0815816A1" is selected, as indicated as 107. Information pertaining to the selected data object 107 is displayed in a data object pane 110. Such information, for example, can include a summary and/or bibliographic information of the selected data object 107 (see the summary tab 114 in the data object pane 110), the text of the selected data object 107, and/or image data relating to the selected data object 107 (see the image tab 112 in the data object pane 110). Other information pertaining to the selected data object 107 is also accessible via the data object pane 110, as further described below.

The contents pane 106 includes navigational buttons 116, 118 which preferably operate relative to the selected data object 107 in the contents pane 106. When selected, button 116 (represented as a down arrow) preferably causes information corresponding to at least the first image page of the next data object in the list of data objects in the contents pane 106 to be retrieved and displayed in the image tab 112 of the data object pane 110. The button 118 (preferably represented as an up arrow), when selected, preferably causes data pertaining to at least the first image page of the prior data object in the list of data objects in the contents pane 106 to be retrieved and displayed in the image tab 112 of the data object pane 110.

According to the image skimming function of the present invention, the first image page of successive documents listed in the contents pane 106 is displayed by repeatedly clicking the navigation buttons 116 and 118. Specifically, to move down the list of data objects displayed in the contents pane 106, the down arrow button 116 is pressed by the user. To move up the list of data objects listed in the contents pane 106, the up arrow 118 is repeatedly pressed by the user.

In an embodiment of the invention, the data objects listed in the contents pane 106 represent patents or documents related to patents (including, but not

limited to, post-issuance documents). In this embodiment, the image skimming feature of the present invention is analogous to a manual search of patents in the shoes of the patent search room, and the collection of documents being searched is analogous to one or more electronic patent shoes. The collection may be one or more groups, for example. During such manual searches, sometimes called "patent shoe searching," practitioners often quickly thumb through the patents in a patent shoe by looking at the first pages of the patents. The image skimming feature of the present invention enables an operator to quickly electronically scan over the first image pages of the patents listed in the contents pane 106, thereby emulating a manual search through the shoes in the patent office search room.

The present invention, however, represents a significant improvement over a manual patent shoe search. These advantages generally pertain to perspective and scope. Specifically, the present invention provides both a global view of the subject matter being searched, and a detailed view of the subject matter being searched. The global view is represented by the list of patents and related patent data objects listed in the contents pane 106. In particular, the contents pane 106 provides the user with an overview of the data objects that is the subject of the analysis.

The detailed view is represented by the data object pane 110, which provides the user with detailed information regarding the currently selected patent from the contents pane 106.

The global view (i.e., the contents pane 106) and the detailed view (i.e., the data object pane 110) are simultaneously displayed for review by the user. Thus, the user can simultaneously electronically view the list of the patents being searched, and the first page of the currently selected patent. Also, the user can immediately modify the focus of the search by selecting any patent listed in the contents pane 106. Such "random access" is not possible with manual patent shoe searching.

Also, the group pane 104 provides an even higher level view of the subject matter being searched.

Further, while conducting the search, the user can easily and effectively create electronic annotations to capture his or her analysis. The annotation features of the present invention are further described below.

FIG. 2 illustrates a flowchart 202 representing the operation of the present invention when performing the image skimming function.

In step 206, the user selects a collection of data objects. The collection of data objects can be selected using any of the features and functions of the present invention described herein. For example, the user can conduct a search wherein the data objects identified by the search represents the collection of step 206. Alternatively, the user can select one or more groups, wherein the data objects contained in the selected groups correspond to the collection of step 206. Still alternatively, the user can select one or more document types, wherein all data objects of the selected document types correspond to the collection of data objects of step 206. A document collection/group can be selected through any combination of searches, attributes, document types, etc., all working together (e.g., "Create a group that consists only of license agreements (document type) containing the text "digital video" (keyword search) whose expiration date is between 1/1/1999 and 12/31/1999 (document attribute)).

In step 208, the data objects in the selected collection are displayed in the contents pane 106.

In step 210, one of the data objects listed in the contents pane 106 is selected. Such selection may be manually performed by the user. Alternatively, by default, one of the documents may be preselected, such as the first document in the list, or the last document in the list. This default operation may be user controllable/adjustable.

In step 212, information pertaining to the selected data object is retrieved. Referring to FIG. 76, an attempt to retrieve such information is first made to

local databases 7612. If image information of the selected document is not contained in the local databases 7612, then the network client 7608 interacts with the enterprise server 7604 to retrieve the image data from the remote databases 7614. Additional details pertaining to the enterprise server 7604 and the network client 7608, as well as other component shown in FIG. 76, is contained in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

In step 214, information pertaining to the selected document is displayed in the data object pane 110. Preferably, information corresponding to at least the first image page of the selected document is displayed in the image tab 112 of the data object pane 110.

In step 216, the user enters a navigational command. Specifically, the user either presses the down arrow button 116 or the up arrow button 118.

If the user pressed the up arrow button 118, then information pertaining to at least the first image page of the prior document listed in the contents pane 106 is retrieved and displayed, as indicated by steps 218, 220, and 222. If, instead, the user selected the down arrow button 116, then information pertaining to at least the first image page of the next data object listed in the contents pane 106 is retrieved and displayed. This is represented by steps 224, 226, and 228. If neither the up arrow button 118 nor the down arrow button 116 was pressed by the user, then the system processes the command that was issued as appropriate, as represented by step 230.

Image skimming is further described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

Context Processing

The present invention supports context processing. A context is a data structure comprising one or more attributes. The context also comprises data objects that satisfy, follow, relate to, and/or have the context's attributes. Contexts are further described below.

Group-Centric Processing

The invention supports a variety of contexts. For example, a group is a context.

As noted above, a group is a data structure that includes a collection of data objects, such as documents. The documents in a group typically (but not exclusively) follow a common theme or characteristic. Groups are typically (but not exclusively based on subject matter. The data objects in a group relate to (or map to) the subject matter associated with the group. For example, a first group may include patents and other documents that map to a product being manufactured and sold by a company. A second group may include patents and other documents that map to a product being considered for future manufacture and sale by a company. A third group may include patents and other documents related to a research product.

Accordingly, groups are a context for organization, visualization, analysis, and other processing.

Groups are further described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

Non-Group-Centric Processing

The invention supports contexts other than groups. Context processing performed by the present invention that does not involve groups is herein referred to as non-group-centric processing.

The invention supports a variety of non-group-centric contexts. For example, a context can be created as a result of processing performed by the system. A context can be created from documents identified by a forward citation tree operation. Similarly, a context can be created from a backward citation tree operation. Another context can be created by searching all groups and/or data object types in the system containing a given document, or having some attribute. In that instance, the context would include all hits from that search.

A particular context supported by the invention is a data object type context. This context is described below.

Data Object Type Processing

According to the invention, each data object has one or more data object kinds. A data object kind represents a classification of a data object. For example, data object kinds include license agreements, non-disclosure agreements, patents, tax forms, personnel evaluations, software programs, user manuals, technical manuals, copyright registration forms, audio clips, movies, multimedia clips, data object links, annotations, and any other data object or document classification of interest to the user.

Data object kinds represent contexts for organization, visualization, analysis, and other processing. For example, it may be useful to analyze all license agreements in which a corporate entity is a party. In this example, the user selects the data object kind "license agreements." All license agreements associated with the corporate entity are selected and may be analyzed by the user. All other information perhaps pertaining to the corporate entity, but not

representing license agreements, are excluded from analysis. Thus, the present invention facilitates the organization, visualization, analysis, and other processing of information of interest to the user.

It is illustrative to note the differences between a group and a data object kind. A group is defined by its attributes, and the documents and other data objects in the group share or follow such attributes. In contrast, a data object kind represents a classification of a data object. The data objects associated with the data object kind may have nothing in common with each other except that they are the same kind of data object.

A given data object, such as a patent, may exist in a plurality of groups. This is illustrated in FIG. 4, where a given patent is contained in a bill of materials group, a user find group, an inventor group, a corporate entity group, etc.. Similarly, a data object may exist in a plurality of contexts. This is illustrated in FIG. 5, for example, where a given data object is included in a group context, an author context, a publisher context, and a topic context.

Corporate Document Browser

FIG. 6 illustrates an example corporate document window or browser 602. Corporate documents are listed in the corporate document window 602. Preferably, corporate documents are defined as non-patent documents. In alternative embodiments, however, corporate documents may include both patents and non-patent documents.

Each corporate document includes a document ID, a title, a data object kind, and a security class. This information is listed in columns 604, 606, 608, and 610, respectively, of the corporate document window 602.

The invention preferably includes various filters to enable users to limit the documents that are listed in the corporate document window 602. The example corporate document window 602 shown in FIG. 6 includes a security

class filter 614 and a data object kinds filter 616. As shown in FIG. 7, the user can select one or more data object kinds for viewing. If, for example, the user selects "contract" in the data object kind filter 616, then only documents having the data object kind of "contract" are listed in the corporate document window 602.

The security class filter 614 allows the user to select one or more security classes. Only documents of the selected security classes are listed in the corporate document window 602. As shown in FIG. 8, for example, if the user selected in the security class filter 614 the class "class 5," then only documents of security class "class 5" are listed in the corporate document window 602.

Preferably, access to documents are restricted/managed/maintained by security information. A user can only access/browse those documents that are consistent with his security information.

The browsers of the invention support inverse functions. An inverse function is one that performs an operation that is the inverse of another operation. For example, browsing a group displays all of the documents in the group. The inverse is browsing all of the groups that contain the document. Many of the browser functions described herein include inverse functions. The inverse functions available for any given browser varies and is implementation dependent.

Data Object Kinds

As noted above, a data object kind is a classification of a document. For example, all license agreements preferably have a data object kind of "license agreement." All non-disclosure agreements preferably have a data object kind of "non-disclosure agreement." All patents preferably have a data object kind of "patent." The level of granularity of the data object kinds is implementation and

application dependent. For example, there may be data object kinds for "U.S. patent," "Japanese patent," "Canadian patents," etc.

A data object may have more than one data object kind. For example, a license agreement may have the data object kind of "license agreement" and "contract."

Typically, the data object kind supported by a system varies over time. Also, typically, different systems will have different data object kinds. Generally, the data object kinds defined and supported by a system correspond to the needs of the users of the system. For example, the data object kinds associated with a patent department may include patent, patent application, license agreement, non-disclosure agreement, and patentability opinion. The data object kinds associated with an accounting department may include payroll statement, invoice, account receivable, license agreement, and budget.

Generally, data object kinds can be created for any criteria or attribute of past, present, or future interest. Data object kinds can be arbitrarily created.

FIG. 9 illustrates a new corporate data object kind window 902, which is displayed when the user elects to create a new data object kind. Preferably, the user enters a name of the data object kind, a description of the data object kind, and an owner of the data object kind. A new data object kind having such name, description, and owner information is created upon pressing the OK button 904.

Security Classes

Preferably, each data object has a security class. Many data objects can have the same security class.

A security class specifies the rights that users have. In other words, a security class specifies user access rights. When a security class is assigned to a document, then access to that document is controlled by the user access rights specified in the security class. Access to the document can be modified by either

changing the access rights associated with the security class, or assigning a new security class to the document. For example, suppose that a security class A gives read and write privileges to a user 1. If document X is assigned security class A, then user 1 has read and write privileges to document X.

FIG. 10 illustrates a new security class window 1002, which is displayed when the user wishes to create a new security class. Preferably, as indicated by the security class tab 1004 in the new security class window 1002, a security class has a name, description, and owner. Referring to FIG. 11, as indicated by the permissions tab 1006, a security class specifies the access privileges of users. A new security class is created when the name, description, owner, and user access information is entered and the OK button 1008 is pressed.

Corporate Documents

In an embodiment of the invention, a corporate document is a data object in the system. It is noted that the following description is applicable to any type of data object (document or otherwise).

FIG. 12 illustrates a new corporate document window 1202 which is used to create a new corporate document. The new corporate document window 1202 includes a file path field. The path to electronic information representative of the document or other data object that is being added to the system as a corporate document is specified in the file path field. This path can be specified via well known browsing techniques, which is initiated by pressing the browse button 1206. This is just one mechanism that serves to identify the location of a document. The invention is not limited to this embodiment. Other mechanisms could be used, such as but not limited to the use of a URL (Universal Resource Locator).

The new corporate document window 1202 also includes a document ID field in which the document ID of the new corporate document is displayed.

Preferably, the document ID is unique among all corporate documents and data objects, and is manually inputted. Alternatively, it is automatically generated by the system.

The new corporate document window 1202 also includes a title field, in which the user can enter a title of the new corporate document. A data object kind and a security class for the new corporate document can be entered in fields 1212 and 1214. If the user wishes to create a new data object kind, the user can press the new button 1216, in which case the new corporate data object kind window 902 (FIG. 9) is displayed. If the user wishes to create a new security class, the user presses the new button 1218, in which case the new security class window 1002 (FIGS. 10 and 11) is displayed. The user can also enter additional bibliographic information pertaining to the corporate document, such as author, source, disclosure date, and publication date. These fields are available in the additional bibliographic fields tab 1222 (See FIG. 14).

FIGS. 13 and 14 illustrate an example where a new corporate document titled "License Agreement Between ABC and EFG" is being created. FIG. 15 illustrates the corporate document window 1302, which displays corporate documents in the system. Note that the new corporate document titled "License Agreement Between ABC and EFG" is listed at 1502.

Data Object Type Databases

Example database tables useful for implementing data object kinds as described above are considered in this section.

FIG. 17 illustrates a type table 1702. The type table 1702 stores information for data object types supported by the system. The type table 1702 includes a record for each data object type defined and supported in the system. Each record of the type table 1702 includes a type_ID attribute corresponding to the unique identifier of the data object type, a name attribute containing the name

of the data object type, a description attribute containing a description of the data object type, and an owner attribute specifying an owner of the data object type.

FIG. 18 illustrates a type document xref table 1802. The type document xref table 1802 stores information that identifies the data objects in each data object type supported by the system. The type document xref table 1802 includes a record for each data object in each data object type supported by the system. Each record of the type document xref table 1802 includes a type_ID attribute that identifies a data object type, and a document_ID attribute that identifies a data object that is in the data object type.

An example of the type table 1702 and the type document xref table 1802 is illustrated in FIGS. 19 and 20. FIG. 19 illustrates an example type table 1902. The type table 1902 includes a data object type called "automation" that has a type_ID of T1. The type table 1902 also includes information pertaining to a data object type called "license agreement" that has a type_ID of T2. FIG. 20 illustrates an example type document xref table 2002. This example type document cross ref table 2002 indicates that contract 1 and contract 3 are of data object type "automation," and contract 1 and contract 2 are of data object type "license agreement."

Context Browser

FIG. 16 illustrates an example context browser 1602, which is similar to the group browser 102 shown in FIG. 1. The context browser 1602 includes a kind pane 1604 in which the data object kinds supported by the system are listed. In some embodiments, the data object kinds are hierarchically organized. For example, a license agreement kind and a non-disclosure agreement kind may be children of a contracts kind. A U.S. patents kind and an Australian patents kind may be children of a patents kind. The hierarchical organization of data object kinds is indicated in the kind pane 1604.

A content pane 1606 list the data objects contained in a data object kind selected in the kind pane 1604. Annotations are listed in an annotations pane 1608 (annotations are further described below). Information pertaining to a data object selected in the contents pane 1606 is displayed in a data object pane 1610. Such information includes any information pertaining to the selected data object, such as summary or bibliographic information in the summary tab 1612, text information pertaining to the selected document in the text tab 1614, image information pertaining to the selected data object in the image tab 1616, audio information associated with the selected document in the audio tab 1618, and video information associated with the selected document in the video tab 1620. Other information pertaining to the selected data object, if any, is also accessible via the data object pane 1610 via additional tabs (not shown in FIG. 16).

FIG. 75 illustrates another example context browser 7502. The context browser 7502 includes a kind pane 7504, a contents pane 7506, and a data object pane 7508. These panes are similar to those shown in FIG. 16.

Operation of Context Processing

Context processing according to embodiments of the invention is illustrated in a flowchart 7702 in FIG. 77. In step 7704, one or more contexts are selected. For example, one or more groups and/or one or more data object types are selected. Each of the selected contexts include any number of data objects.

In step 7706, the data objects in the selected contexts are processed. Such processing includes any combination of the functions described herein, and/or the functions described in U.S. Patent Applications titled "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-centric and Group-oriented Data Processing," Ser. No.

08/921,369, and/or "System, Method, and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, all of which are herein incorporated by reference in their entireties. Such processing may be automatic, manual, or combinations thereof.

Data Object Preview

FIG. 76 illustrates a block diagram of an embodiment of the present invention. An enterprise server 7604, which is also referred to as an intellectual property asset manager (IPAM) and a business decision system (BDS), performs processing on data objects as described herein. Information on such data objects is stored in IPAM databases 7614. Such information includes bibliographic information, textual information, image information, video information, audio information, and other information associated with the data objects.

Preferably, users interact with the enterprise server 7604 and the IPAM databases 7614 via network clients 7608 and web clients 7606. Network clients 7608 interact directly with the enterprise server 7604 (although there may be networks between the network clients 7608 and the enterprise server 7604). Web clients 7606 interact with the enterprise server 7604 via web server 7610.

Preferably, network clients 7608 include local databases 7612. Web clients 7606 may also include local databases. The local databases 7612 store information on data objects. Preferably, the information contained in the local database 7612 is a subset of the information contained in IPAM databases 7614. For example, the local database 7612 may only include information on a limited number of data objects. Also, for any given data object, the local database 7612 may only store some information pertaining to the data object. For example, the database 7612 may only store bibliographic or summary information pertaining to the data object. The database 7612 may not store more detailed information

pertaining to the data object, such as textual information, image information, video information, audio information, etc.

Further information pertaining to the enterprise server 7604, the network client 7608, the web client 7606, the web server 7610, and databases 7614 and 7612 are found in U.S. Patent Application "System, Method, and Computer Program Product for Patent-centric and Group-oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

FIG. 21 illustrates a browser 2102 (also called a console), which may be a group browser or a context browser. As described above, the contents pane 2108 lists documents that are contained in the group or kind selected in the group/kind pane 2106.

The data object pane 2104 illustrates information pertaining to the document selected in the contents pane 2108. Preferably, the data object pane 2104 displays summary or bibliographic information pertaining to the document selected in the contents pane 2108. This is the case in the example FIG. 21.

The user can elect to display other information pertaining to the selected document by selecting another tab in the data object pane 2104, such as the image tab 2110 shown in FIG. 21. However, such additional information pertaining to the selected document may not be present in the local database 7612. The invention supports a document or data object preview function for addressing such instances.

Specifically, in cases where requested information pertaining to a document or other data object selected in the contents pane 2108 is not present in the local databases 7612, the invention displays a preview button 2106. When the user presses the preview button 2106, the network client 7608 or the web client 7606 interacts with the enterprise server 7604 to retrieve at least a portion of the additional information pertaining to the selected data object. Preferably, the text of the selected document is transferred from the IPAM databases 7614

to the network client 7608 or the web client 7606. This text is then displayed in a preview window 2202 for review by the user (see FIG. 22).

In other embodiments, additional information is transferred from the IPAM databases 7614 to the network client 7608 or the web client 7606. Such information may include image information, audio information, video information, or any other information pertaining to the selected data object that is stored in the IPAM databases 7614. As shown in FIG. 22, such information can be displayed by selecting the appropriate tab in the preview window 2202.

In some embodiments, if the requested information on the selected document is not present in the IPAM databases 7614, the enterprise server 7604 operates to obtain the information from an external source. For example, the enterprise server may interact with a third party information provider, such as Lexis-Nexis, or may interact with some other source, such as a human operator, a company database (such as a personnel database, a finance database, etc.), etc.

The user reviews the information in the preview window 2202. Preferably, this information is available only on a temporary basis, and it is not stored in the local databases 7612. If the user wishes to store the information in the local databases 7612, then the user presses an order button 2206 or issues a similar command. In response to the selection of the order button 2206, the enterprise server 7604 retrieves any additional information pertaining to the selected document from the IPAM databases 7614 or from some other source, and transfers such information to the network client 7608 or the web client 7606. This information, as well as information previously provided, is stored in the local databases 7612.

The processing pertaining to the data object preview function of the present invention is illustrated in FIG. 3. Specifically, FIG. 3 illustrates the interaction between the network client 7608 and the enterprise server 7604 when performing the data object preview function 318. The network client 7608 sends a preview request message to the enterprise server 7604. The preview request

message is sent by the network client 7608 to the enterprise server 7604 when the user pressed the preview button 2106. The preview request message includes information that identifies the data object selected in the contents pane 2108. The preview request message also indicates the type of data that is requested. For example, in some embodiments, only text data pertaining to the selected data object is requested. In other embodiments, other types of data is requested, such as image data, audio data, video data, or any other or all data pertaining to the selected data object. In other embodiments, a given type of data is always requested and provided, such as textual data, in which case a preview request message does not include an indication of the type of data that is requested.

Optionally, the network client 7608 and the enterprise server 7604 perform a financial exchange whereby funds are transferred from the network client 7608 to the enterprise server 7604. The financial exchange is performed in those cases where the preview function is available only as a pay-for-used service. In the financial exchange, the funds are electronically or otherwise transferred from the network client 7608 to the enterprise server 7604. The invention also includes other economic models, such as but not limited to an unlimited service plan, where for a given fee the user can download any number of documents, and a hybrid plan where for a fee the user can download a given number of documents. Once this limit is exceeded, the user pays on a per-document rate.

The enterprise server 7604 retrieves from the IPAM database 7614 the requested data pertaining to the selected data object, and transfers this data to the network client 7608. The IPAM database 7614, in one embodiment, obtains such data from a data source 304. The data is then processed (indicated by oval 306), as necessary, and then stored in the IPAM database 7614. The data source 304, for example, may be a Patent Office, departments of a corporate entity, third party information providers (as discussed above), etc., and/or may result from

data processing performed by automatic means, manual means, or combinations thereof.

Upon receipt of the preview data by the network client 7608, the network client 7608 displays the preview data in the preview window 2202, as described above.

The operations performed by the present invention when the user requests the retrieval of a data object is also represented in FIG. 3 as a data object retrieval function 320. In accordance with such processing, the network client 7608 transfers a document request to the enterprise server 7604. The document request was generated by the network client 7608 when the user pressed the order button 2206 in the preview window 2202 (FIG. 22). The document requests includes information that identifies the selected data object in the contents pane 2108 and also identifies the type of data that is requested (i.e., text, image, audio, video, etc.). In cases where the data object request functionality is offered only as a pay-per-use service, the network client 7608 and the enterprise server 7604 perform a financial exchange operation in which funds are electronically or otherwise transferred from the network client 7608 to the enterprise server 7604. The enterprise server 7604 retrieves the requested information from the IPAM database 7614 or other source, and transfers such information to the network client 7608. The network client 7608 then preferably stores this information in its local databases 7612 (FIG. 76) for later access by the user.

Annotation

The present invention supports a variety of annotation types. An annotation type is defined by two attributes. First, by whether annotations of the annotation type are attached or linked to data objects, or portions of data objects. Second, by the scope of annotations of the annotation type. The scope defines the extent to which the annotations are visible or accessible. FIG. 23 is a table

2302 that summarizes the attributes of the annotation types of the present invention.

Annotations are further described in U.S. Patent Nos. 5,623,681; 5,623,679; and 5,806,079, all of which are herein incorporated by reference in their entireties.

Document Annotations

A document annotation is attached or linked to a selected portion of a data object contained in a group. The scope of a document annotation is the group in which the linked data object is contained, and that was active or selected at the time that the document annotation was created. Document annotations are accessible only when their respective groups are active.

A document annotation may be linked to multiple data objects. Also, a document annotation may be unlinked from any or all linked data objects. When a document annotation is unlinked from all linked data objects, it is called a floating, free, or unlinked annotation (as opposed to a linked annotation). Unlinked annotations maintain their original scope, and operate just like linked annotations, except that they have no links to data objects to follow. Unlinked annotations can be linked to data object portions through appropriate user command. It is noted that this description pertaining to document annotations is also applicable to other linkable annotation types described herein.

For example, assume that a document annotation is linked to a document D1. Document D1 is contained in groups G1 and G2. However, at the time that the document annotation was created, group G1 was active. That is, group G1 was selected in the group pane 104 of the group browser 102 (see FIG. 1) at the time that the annotation was created. In this case, the document annotation is linked to document D1, and has a scope of group G1. The document annotation

is only accessible when group G1 is active. The document annotation is not accessible when group G2 is active, even though document D1 is in G2.

Patent Annotations

The invention supports a variety of types of document annotations. For example, the invention supports patent annotations. Consider FIG. 24, which displays a summary of U.S. Patent No. 5,668,742 in a data object pane 2406 of browser 2402. Upon appropriate user command, the text of U.S. Patent No. 5,668,742 is displayed in a text window 2502 shown in FIG. 25. According to the invention, a patent annotation is created by selecting a portion of a patent, such as the selected portion 2504 shown in FIG. 25. Such operation results in creating a subnote 2509 in a note window 2508 that is linked to the selected portion 2504 of U.S. Patent No. 5,668,742.

User comments can be entered into the subnote 2509. In an embodiment, location information 2511 is stored and displayed proximate to the subnote 2509 (this is applicable for all annotation types described herein). Such location information 2511 indicates the location of the selected portion 2504 in the data object, *i.e.*, U.S. Patent No. 5,668,742.

Notes are listed in the annotation pane 2510. For example, in the current example, the newly created note is listed in the annotation pane 2510 as entry 2512.

Corporate Document Annotations

The invention also supports corporate document annotations, in which any type of document can be annotated. For example, in FIG. 28, summary information pertaining to a corporate document selected in a contents pane 2404 is displayed in the data object pane 2406. Image information pertaining to the

selected document is displayed in an image window 2902 upon execution of appropriate user commands, as shown in FIG. 29. As indicated in FIG. 30, a portion 3002 of the image in image window 2902 has been selected. This operation causes a subnote 3006 to be created in a note window 3004. The subnote 3006 is linked to the selected portion 3002. The user can enter information or comments pertaining to the selected portion 3002 in the subnote 3006 (that is, the user can annotate the selected corporate document).

Image Annotations

The invention supports image annotations, wherein images can be annotated. For example, in FIG. 26, image information pertaining to U.S. Patent No. 5,668,742 is displayed in the data object pane 2406 of the browser 2402. Upon appropriate user command, such image information is displayed in larger form in an image window 2702 shown in FIG. 27. As indicated in FIG. 27, the user has selected a portion of the image. This selected portion is indicated as 2704. Upon such selection, a subnote 2708 is created in a note window 2706. This subnote 2708 is linked to the selected portion 2704. The user can enter comments pertaining to the selected portion 2704 in the subnote 2708 (that is, the user can annotate the selected portion 2704).

Group Annotations

A group annotation is not attached to a data object or a portion of a data object. The scope of a group annotation is the group that was selected or active at the time that it was created, and the group annotation is linked to the group. A group annotation is visible only when its associated group is selected or active. Preferably, any user with access to a group has access to the group annotations associated with the group.

Data Object Type Annotations

Data object type annotations are similar to group annotations. Data object type annotations are not attached to data objects. Instead, data object type annotations are linked to data object types. The scope of data object type annotations is the associated data object type that was specified when the annotation was created. Data object type annotations are only accessible when their associated data object types are selected or active. Like group annotations, access to data object type annotations is generally open to any user who has access to the associated data object type.

Case Annotations

A case is a data structure that includes a plurality of data objects. Such data objects may be in one or more groups, one or more data object types, and/or one or more other cases. A case may span multiple groups, data object types, other cases, etc. That is, a case can be set to include a given set of groups, data object types, cases, etc. A case is a type of context.

A case annotation is attached or linked to a portion of a data object contained in a case. The scope of the case annotation is the case that was active or selected when the case annotation was created. Case annotations are only visible if the associated case is selected or active. Access to case annotations is preferably under the control of the case owner. In an embodiment, a document that is being annotated is added to the active case automatically.

Enterprise Annotations

Enterprise annotations are attached and linked to data objects. The scope of an enterprise application is the data object to which it is linked. Accordingly, an enterprise annotation is accessible whenever the data object is accessible. Therefore, whenever a data object is contained in the active context, the enterprise annotations linked to that data object are accessible. In an embodiment, access to enterprise annotations is limited to trusted individuals, such as administrators or administrator designees, given the wide scope of enterprise annotation

In an embodiment, the ability to create an enterprise annotation is granted to the user by the system administrator. The administrator would specify the types of data objects that can be annotated by a user (e.g., NDAs, patents, license agreements, and so forth).

In an embodiment, enterprise annotations are only visible to a given user if the creator of the enterprise annotations exported his or her annotations to the given user, and the given user subscribed to the enterprise annotations of the creator. For example, if user 1 created an enterprising annotation 1 linked to data object 1, then user 2 when viewing data object 1 would only have access to enterprise annotation 1 if user 1 exported his enterprise annotations to user 2, and user 2 subscribed to the enterprise annotations of user 1. This exporting/subscription feature of the invention operates as both a security feature (enabling a creator to limit access to only those people to which his enterprise annotations have been exported to), and a filtering tool (enabling a user to display only those enterprise annotations generated by creators to which the user has subscribed).

User Interface and Note Processing

As noted above, the annotations pane contained in the browser or console window lists the currently active annotations. An annotation is active if its context is active. For example, a document annotation is active if it is linked to a data object contained in the active group (*i.e.*, the group selected in the group pane of the browser). A case annotation is active if its associated case is active (that is, its associated case is selected). A group annotation is active if its associated group is active (that is, its associated group is selected in the group pane of the browser). A type annotation is active if its associated type is active (that is, its associated type is selected in the type pane of the context browser). An enterprise annotation is active if the data object to which it is linked is contained in the active context (that is, it is contained in the active group or the active data object type).

Embodiments of the invention support a variety of formats for the annotation pane. Such formats are presented in the figures. For example, FIG. 31 illustrates an annotation pane 3102 having a tab for each annotation type. Specifically, there is a tab 3106 for document annotations, a tab 3108 for case annotations, a tab 3110 for group annotations, a tab 3112 for type annotations, and a tab 3114 for enterprise annotations. To view the active annotations of any given type, the user selects the tab associated with the type. For example, to view the currently active enterprise annotations, the user selects the enterprise tab 3114.

In embodiments of the invention, different types of annotations are denoted by different types of attributes. For example, different annotations may be displayed using different display attributes, such as font, color, size, underlining, icons, etc.

It is noted that the annotation type supported by the invention represent additional types of contexts. For example, enterprise annotations represent a type

of context that include either the active enterprise annotations, and/or the data objects linked to the active enterprise annotations.

FIG. 32 illustrates a flowchart 3202 illustrating the operation of the invention when creating an annotation that linked to a data object (that is, when creating a document annotation, a case annotation, or an enterprise annotation).

In step 3206, the user selects an annotation mode. This can be done, for example, by selecting an annotation pen 3001 (see, for example, pens 3001 in FIG. 30). It is noted that pens can be assigned to different types of annotation types such that the type of annotation to be created depends on the pen that is selected. Alternatively, this can be done by selecting an appropriate menu command, such as a drop down menu or a menu that it appears upon pressing the right mouse button.

In step 3208, the user selects a portion of a data object using the selected annotation pen, if any.

In step 3210, the user selects an annotation type (*i.e.*, a document, case, or enterprise annotation). Its noted that step 3210 is performed only when the user has not previously selected an annotation type. For example, as noted above, the user may have previously selected an annotation type when he selected the annotation pen 3001.

In step 3212, the system creates a new annotation of the type selected in step 3210, and links the annotation to the portion of the data object selected in 3208. Further in step 3212, the user enters comments or notes to annotate the selected data object portion.

In step 3214, information pertaining to the new annotation, including the user entered comments, the linking information, etc., is stored in an annotation database.

FIG. 33 illustrates a flowchart 3302 that represents the operation of the invention when creating a group or a data object type annotation (that is, when creating an annotation that is not linked to a data object).

In step 3306, the user selects a group or a data object type. The selected group or data object type becomes the active group or data object type.

In step 3308, the user selects a group or type annotation mode. This can be done, for example, by selecting an appropriate menu command, or by selecting an appropriate annotation pen 3001.

In step 3310, the system creates a new annotation of the selected type. The new annotation is linked to the active or selected group or data object type. Also in step 3310, the user enters notes or comments to thereby annotate the active group or data object type.

In step 3312, information pertaining to the annotation, such as the user's comments, the linking information, etc., are stored in an annotation database.

Form-based Annotations

As noted elsewhere, the present invention supports annotation of data objects. For example, FIG. 78A illustrates an example window 7802 wherein a data object 7803 is being displayed. The present invention enables annotation of any portion of the data object 7803.

The window 7802 includes a plurality of annotation pens 7804. The user places the system in an annotation mode by selecting one of the annotation pens 7804 (the invention includes other ways of placing the system in the annotation mode, such as selecting an appropriate command from a menu or dialog). The user utilizes the selected annotation pen to select a portion of the data object 7803. For example, FIG. 78B illustrates a selected portion 7806 of the data object 7803 that has been selected using the selected annotation pen. Upon selecting the selected portion 7806, the system automatically creates a new annotation, which is illustrated as note segment 7810 in note 7808. The note segment 7810 is automatically linked to the selected portion 7806.

Preferably, the cursor is automatically positioned in an annotation field 7814 of the note segment 7810 so the user can record his observations. When finished, the user presses the save button 7850. Upon pressing the save button 7850, information pertaining to the annotation, such as the user's comments in the annotation window 7814, linking information, location information, the name of the annotation, whether the annotation includes privileged information, etc., are stored in an appropriate data format. Preferably, in a single user system, such annotation information is stored on the local computer. In a multi-user system, the annotation information is preferably stored in a database in such a way that the information can be securely shared with other users.

In the embodiment being described, the system does not provide any guidelines or place any restrictions on the scope and content of the comments. Accordingly, comments entered by the user may be of no value to himself or others. In some cases, the user's comments may be inappropriate given the circumstances under which the comments were solicited.

The user's comments comprise free form text which is entered in the annotation field 7814. In the embodiment being described, the comments are stored in an unstructured data format that is appropriate for storing free text, such as but not limited to a flat file database or a binary large object (BLOB) in a relational database. This data format is not structured in any way and is therefore not appropriate for storing in a standard relational database format.

According to an embodiment, the invention supports form-based annotations. Form-based annotations extend the operation of the annotation system described above to allow an expert user to create a specific form that is directed and limited to the information that one wishes the user to provide. Herein, for reference purposes, the person who creates a form is referred to as the form creator and the person who uses the form in the process of making an annotation is the annotation creator. Preferably, the form-based annotation system operates in a secure, multi-user, client/server system.

FIG. 79 illustrates an example annotation form, which is called the screening form 7902. According to the invention, annotation forms include any well known input widgets, such as radio buttons (choose one from many items), check boxes (which represents, for example, yes/no or true/false), text input fields (that may include edit checks), etc. For example, the annotation form 7902 includes radio button fields 7906, 7908, 7912, a structured date field 7910, and a name field 7904 which includes an edit check 7914 to ensure that proper data is entered.

According to the invention, data entered into one or more fields of an annotation form are stored in relational database fields. The number of annotation form fields that are stored in relational database fields vary from form to form, depending on the specific implementation of the form creator.

In the example of the annotation form 7902 of FIG. 79, information entered into the name annotation form field 7904 is stored in a name field 7926 of a relational database table 7922. Information from the radio button fields 7906, 7908, and 7912 are stored in fields 7928, 7930, and 7936, respectively, of the relational database table 7922. Date information entered into the when annotation form field 7910 is stored in a date field 7932 of the relational database table 7922.

FIG. 80 illustrates another example annotation form, called the comments form 8002. In the example embodiment of FIG. 80, information entered in the comments form 8002 is stored in either a structured table 8016 or a free form table 8018. The structured table 8016 and the free form table 8018 collectively comprise a comments form database 8014. In particular, name information entered into a name annotation field 8004 is stored in a name field 8022 of the structured table 8016. Information from radio button fields 8006 and 8008 are stored in a product 1 field 8024 and a product 2 field 8026, respectively, of the structured table 8016. Free text information entered into a comments on claim

1 field 8010 and a comments on claim 2 field 8012 are stored in rows of the free form table 8018.

It should be understood that the particular database table implementations of the screening form 7902 and the comments form 8002 shown in FIGS. 79 and 80 are provided for purposes of illustration only, and not limitation. Other database and table implementations will be apparent to persons skilled in the relevant arts based on the teachings contained herein. Also, the particular configurations and arrangements of the screening form 7902 and the comments form 8002 are provided for illustrative purposes only, and are not limiting. Other forms will be apparent to persons skilled in the relevant arts based on the teachings contained herein.

FIG. 82 illustrates a flow chart 8202 that represents the operation of the form-based annotation system according to an embodiment of the invention.

In step 8204, a form creator creates a form, such as the screening form 7902 or the comments form 8002. The form creator is any user who has been designated as a person capable of creating and editing forms.

In step 8206, one or more relational database tables are created for the form, if necessary. Information entered into the form created in step 8204 will be stored in these database tables. These database tables may be created by the form creator or by some other user, such as but not limited to a system administrator.

In step 8208, the form creator selects one or more relational database tables for storing the information that is to be entered into the form. The selected database tables may include, for example, one or more of the tables created in step 8206, and/or other database tables created at other times.

In step 8210, the form creator assigns one or more fields (columns) from the selected relational database tables to each of the fields of the form. Data entered into the fields of the form are stored in the selected database table fields.

In step 8212, the form creator specifies and defines any additional features of the form. For example, the form creator can specify processing associated with any of the fields of the form. Considering FIG. 79, for example, the name field 7904 involves processing 7914 to check name information against a company directory to ensure that the name information was entered correctly into the name field 7904. The screening form 7902 also includes processing 7916 associated with the when field 7910 that involves confirming that the date entered into the when field 7910 is a valid date. The when field 7910 includes further processing 7920 associated with comparing the date entered into the when field 7910 with a projected IDS (information disclosure statement) filing date 7918 to identify issues associated with submitting references to the U.S. Patent Office. For example, if the projected IDS filing date 7918 is greater than 3 months after the date entered into the when field 7910, then a flag is set in database field 7934. The flag indicates that there may be an issue relating to whether or not the reference can be cited with the U.S. Patent Office in satisfaction of the duty of disclosure for a pending U.S. patent application. Other form features that can be specified in step 8212 is further described below.

After step 8212 is completed, the design of the form is complete. Steps 8214, 8216, and 8218 involve the manner in which an annotation creator uses a form.

In step 8214, the annotation creator selects a form. The form is tied to a specific pen (or some other annotation invocation mechanism, such as an option on a menu). The annotation creator selects the pen for the form, thereby informing the system which form to invoke, how to collect/store the data, etc.

In step 8216, the selected form is displayed. The annotation creator enters information into the form as specified by the fields of the form.

In step 8218, information entered into the fields of the form is stored in the associated relational database fields as defined by step 8210.

The operation of the form-based annotation system of the present invention is further described with reference to a flow chart 8302 shown in FIGS. 83A and 83B. Steps 8304, 8306, and 8308 of flow chart 8302 generally correspond to steps 8204, 8206, 8208, 8210, and 8212 of flowchart 8202 of FIG. 82. Step 8310 of flowchart 8302 generally corresponds to steps 8214, 8216, and 8218 of flowchart 8202 of FIG. 82.

In step 8304, preferably the system administrator authorizes a given user (the form creator) to create forms. In other words, the ability to create, modify, edit, and otherwise operate with forms is preferably a secured operation. This is because the creation of the form includes appropriate provisions for storing the data acquired for the use of the form in relational database(s) for backend statistical processing, as well as other processing.

In step 8306, the form creator logs into the system and creates a form to capture the data he is interested in capturing.

The invention supports all known input widgets, including radio buttons, text input fields, check boxes, etc. The form creator explicitly or implicitly associates input widgets with back end database tables. Even the meta content of the tables may be stored with the form allowing for the use of sophisticated data mining applications of the form and the data. It is also possible to associate programmic operations with the form. Examples of such operations include edit checks on input fields, processing based on data entered into fields, invocation of other applications based on information entered into fields, etc.

In step 8308, the form creator associates the form with a pen (such as one of the annotation pens 7804 as shown in FIG. 78A) or other annotation mechanism (such as a pull down menu). This is part of a publishing process in which the form creator also specifies appropriate identifying characteristics of the form (such as a name of the form) and the storage locations of the data that is to be captured by form (that is, the relational database table fields in which the data entered into the form is stored).

The publishing process also allows the form creator to specify which user or groups of users should be allowed access to the form. Users are preferably identified by either their log in I.D. (identification) or by a role I.D. that specifies the role that they are assuming with respect to the operation of the system. In this way, the form creator can customize the operation and presentation of a given form-based on the user's role. For example, for a particular form, some fields are shown to marketing people, whereas other fields are shown to engineering people. Some fields may be shown to both marketing and engineering people. Further, some processing may be performed when the user has a marketing role, and other processing may be performed when the user has an engineering role. Some processing may be shown to both engineering and marketing people.

In an embodiment, the role based process described above allows a person to play multiple roles at the same time. Thus, a marketing/engineering role would see both marketing and engineering forms.

The form creator can also associate multiple forms with a single annotation mechanism. The form creator can further decide if they want the system to route the form to specific users and, if so, the manner of routing.

After completion of step 8308, design of the form is complete. In step 8310, the annotation creator uses the form to create an annotation. In particular, the annotation creator creates a new annotation by possibly using a pen, as described above. The system invokes the form associated with the pen and presents it to the user. In a case that multiple forms have been associated with a single annotation mechanism, the system presents a choice of forms to the user for selection (e.g., through a pull down menu, or pick list). Alternatively, the system sequences through the forms associated with the annotation mechanism. The user enters the information called for by the form, and the system stores this information in the appropriate relational database locations (and potentially other

non-relational database locations) as specified when the form was being designed.

The form-based annotation system will now be further described by considering a number of examples.

Medical Example

As noted above, form-based annotations can be used to better ensure that specific and desired information is collected from the user. Consider a doctor who has asked a colleague to review an X-ray to determine if a specific medical condition is present. The colleague may not fully understand the request, and enter irrelevant information as an observation if they are using a text field for annotations, as with the "free-text" annotation system of FIGS. 78A and 78B. However, the doctor could create a form that asks specific medical questions. In this case, when their colleague makes an annotation, he would be prompted for the desired medical input.

Legal Example

A lawyer may wish to have an engineer comment on one or more claims of a patent, but would like to control the specific commentary that can be entered by the engineer. With free-text annotations, the engineer could inadvertently enter inappropriate information. With form-based annotations, the lawyer could limit the amount and kind of information that an engineer could enter.

For example, consider the scenario shown in FIG. 81. A novice user 8108 fills out a screening form 8106 (see FIG. 79), which is then stored in the screening form database 7922. The completed screening form 8106 is also automatically routed to the IP (intellectual property) attorney 8110 (this operation is programmed into the screening form). If, in completing the

screening form 8106, the user 8108 checked the "Do you believe we should further study this patent?" field, then the comments form (FIG. 80) is routed to the novice user 8108 and an expert user 8120 to solicit their comments (this operation is programmed into the screening form).

The comments form is completed by both the novice user 8108 and the expert user 8120. Their completed comments forms 8118, 8124 are stored in the comments form database 8014. Also, the novice user's completed comments form 8118 is routed to the expert user 8120 for review (this is programmed into the comments form). The expert user 8120 may revise the novice user's completed comments form 8118. The revised comment form is indicated as 8122. The revised comment form 8122 is stored in the comments form database 8014.

Similarly, the completed comments forms 8118 and 8124 are routed to the IP attorney 8110, who may enter his own comments 8112, which are stored in the comments form database 8014 (this is programmed into the comments form).

Product Development Example

There are times when one user, who is an expert in one domain, wants specific opinions from one or more users whose expertise lies in different, but related, domains. In this case the first expert could annotate a specific portion of a data object. When another expert accesses this annotation the system will prompt them, through the use of a form-based annotation, for their opinion of the annotated data object. The system provides a wide variety of mechanisms to obtain feedback from expert users, as well as other selected users, including routing and multiple routing slips and time-based expirations.

Consider a product development example where a product marketing manager may want an engineer to provide a cost estimate on how much it would

cost to manufacture a patented product to help determine long-term product strategy. In this case, the marketing manager would associate appropriate edit-checks on the input widgets of the form to ensure that the engineers are entering in numeric information in a reasonable range. Because a decision is needed quickly the marketing manager routes it through the company email system to all of "ENGINEERING" specifying a time-limit of 7 days. If an engineer does not respond within 7 days the system automatically removes the request from their "work queue".

Note that an engineer may annotate the patent several times, thereby providing multiple cost estimates, one for each annotated portion of the patent that would, in the opinion of the engineer, result in a component or portion of the product. In this case, the system would operate in a manner similar to free-text annotations, automatically creating sub-forms contained within a larger "form" (much like creating sub-notes in the context of a note). The form-creator would have to decide on the operation of these forms during the creation process (e.g., the form-creator would either allow or reject the use of multiple sub-forms within a form).

Additional Product Development Example

Form-based annotations enable the collection and storage of observations of users who make annotations in a way that these observations can be used for reliable reporting and other statistical data mining operations.

Consider the product development example discussed above. After 7 days the product marketing manager could run a report that produces the weighted average of the cost estimates from the engineers. The system could perform this report by querying specific database columns, and extracting data from specific database columns. For example, the system can execute a search in a Department column of the relational database to identify all rows of the

database that correspond to annotations made by members of a given engineering department. The system can then extract data from the "Cost Estimate" column of the identified rows.

Accordingly, the form-based annotation system of the invention facilitates data mining, statistical processing, and other processing of the annotation data.

If, instead, a free text format was used, it would be necessary to parse the free-text to extract the desired information.

It is noted that the invention is not limited to only storing information in formally structured fields within a relational database. In fact, a form could contain a mixture of structured fields (such as dates, numbers, or Boolean) and unstructured text fields, as discussed above.

Medical Teaching Example

With free text annotations, for example, the user selects a portion of a data object and then describes why the selected portion is relevant. Sometimes this is exactly the opposite order of what is desired. Specifically, sometimes an expert user wants a different (and perhaps more novice) user to annotate a portion of a data object in response to what the expert user created.

For example, a teacher could create the contents of an annotation and ask students to "find the spot where the annotation is true". For example, a teacher could prepare a series of x-rays of difficult to diagnose fractures. The assignment could be: "Identify the location of the hairline fracture". Students would then annotate the data object where they thought the fracture was using a form-based annotation, possibly adding commentary on their selection in a new annotation window (or in the original). The teacher could then compare the work of the student to a known standard.

Searching

The invention supports a variety of types of data objects. For example, the invention supports patent related data objects, corporate documents (which, in some embodiments, constitute non-patent related data objects), etc. Such data objects are preferably stored in system databases, such as the IPAM databases 7614 and/or local databases 7612 (see FIG. 76). The invention also supports data objects that are not stored in system databases. For example, the invention supports data objects that are available from third parties. Such third parties include, but are not limited to, on-line data providers, such as LEXIS-NEXIS, Dialog, Westlaw, Derwent, etc. Third parties also include, but are not limited to, public or government databases, such as those associated with the SEC, FCC, FDA, Department of Motor Vehicles, Social Security, etc. With regard to such third party providers, the invention supports establishing a link with the third parties, and enabling users of the invention to interact with the third parties via the link. Such third party providers include well known interfaces or APIs (application programming interfaces). In embodiments, the invention interacts with such third parties via their respective interfaces or APIs.

The invention includes functionality to enable users to conduct complex searches through the data objects that it supports. The invention provides detailed graphical user interfaces (GUI) to enable users to define their searches. FIG. 34 illustrates an example search GUI 3402. In the example of FIG. 34, the search GUI 3402 includes a variety of tabs associated with the data object types supported by the invention.

For example, the invention includes a patent documents tab 3404 for specifying a search among the patents stored in the system databases. The search GUI 3402 also includes a corporate documents tab 3406 for specifying a search among the corporate documents stored in the system (see FIG. 35). The search

GUI 3402 also includes an all documents tab 3408 for specifying a search among all data objects stored in or supported by the system (see FIG. 36).

The search GUI 3402 further includes tabs for third party information providers supported by the system. For example, and without limitation, a LEXIS-NEXIS tab 3410 in the search GUI 3402 is selected to define a search to be performed by the LEXIS-NEXIS system. In an embodiment, upon providing such search criteria, a link is established with the third party information provider wherein the third party is instructed to perform the specified search. Search results are then returned to the system (i.e., the enterprise server 7604, the network client 7608, and/or the web client 7606). Such information is then treated as any other system information.

Further details pertaining to the searching capabilities of the invention are described in U.S. Patent Application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

Data Presenting and Processing Using Visualization Technologies, Such as Hyperbolic Trees

The invention supports the use of a variety of visualization technologies to present information to users. Such technologies include, but are not limited to, hyperbolic trees. Other technologies useful for presenting data to users are discussed in sections below.

A hyperbolic tree or a hyperbolic browser is a well known "focus plus context" technique for visualizing and manipulating data hierarchies, such as trees. A hyperbolic browser assigns for display a portion of a tree while still embedding in it the context of the entire tree. The essence of this scheme is to lay out the tree in a uniform way on a hyperbolic plane and map this plane onto a display region, such as but not limited to a circular display region (other

display geometries are also possible). Thus, hyperbolic trees support a smooth blending between focus and content, as well as continuous redirection of the focus.

An example hyperbolic tree 3702 is illustrated in FIG. 37. The hyperbolic tree 3702 includes a root node 3704 and a plurality of additional nodes connected either directly or indirectly to the root node 3704. Additional portions of the hyperbolic tree 3702 can be expanded and displayed in the window 3712 (that is, focus can be redirected to those portions) by traversing to the desired portions in the hyperbolic tree 3702. An operator traverses the hyperbolic tree 3702 using a pointing device, such as a mouse.

According to the invention, the hierarchical information represented in hyperbolic trees may come from any source, including but not limited to patent offices, corporate entities, public records, individual users, etc. In an embodiment, the hierarchical information is patent related, but the invention is not limited to this embodiment.

The invention provides a variety of functions to enable users to customize the display of hyperbolic trees. For example, embodiments of the invention store state information in each node of the hyperbolic tree.

FIG. 42B illustrates example node state information 4202 which is stored in each node of the hyperbolic tree. The node state information 4202 includes the label that is displayed in the associated node. For example, the label of node 3706 in FIG. 37 is "claim 271."

The node state information 4202 also includes display attribute information that indicates the manner in which the node is to be displayed in the window 3712. Such display attributes include any attributes for displaying information on a computer screen, such as font, color, bold, italics, underlining, visible/invisible, etc. The display attributes also include user defined and custom attributes, which are entered by a user. The user defined attributes are useful to

enable users to specify the display of nodes to meet their specific applications.

In some embodiments, the display attributes of a node are automatically set based on the information corresponding to the node. For example, when a hyperbolic tree is used to display a patent claim tree, the display attributes of nodes may depend on whether the nodes correspond to a patent or a claim, an independent claim or a dependent claim, a disclaimed claim, a valid claim, an invalid claim, an infringed claim, a non-infringed claim, a claim that has been annotated, etc.

In some embodiments, the display attributes may be set according to information entered into the user defined field. For example, if the user enters that a given claim is invalid, then the font and color display attributes may be set to preselected values corresponding to invalid claims. Therefore, the user defined field(s) and the display attributes portion of the node state information 4202 may specify actual display attributes (such as crosshatching), or state information that the system uses to automatically set the display attributes.

The node state information 4202 also includes link information that establishes a link between the node and associated information in the system databases (i.e., the IPAM databases 7614 and/or the local databases 7612 shown in FIG. 76). In the case of a claim tree 3702 in FIG. 37, the link information establishes a link between the node and the text of the claim associated with the node. Alternatively, such claim text can be stored in the node itself as part of the node state information 4202. In other embodiments, a portion of the claim text (such as the preamble or a portion thereof) is stored in the node as part of the node state information 4202, and the link information in the node state information 4202 establishes a link to all of or the remaining portion of the claim text.

The invention supports functions to edit the appearance of a hyperbolic tree. For example, the invention allows a user to prune a tree by removing nodes

or branches. In embodiments of the invention, such pruning is accomplished by deleting the identified nodes and/or branches from the tree. In other embodiments of the invention, such pruning is accomplished by appropriate setting of the display attributes of the selected nodes and/or branches. Specifically, nodes corresponding to portions of the tree which a user has selected to delete are set to invisible or transparent.

Specific embodiments of hyperbolic trees supported by the invention are described below. It is noted that these embodiments are provided for illustrative purposes only, and are not limiting. Additional applications of hyperbolic trees will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

Patent Claim Trees

As noted above, the invention includes a patent claim tree module that supports patent claim trees.

The patent claim tree module generates a graphically displayed claim tree from a collection of claims. The patent claim tree module works both as a standalone product, and as an add-on to IPAM.

The patent claim tree module includes a mechanism to elicit help from the user when it fails to understand enough about a claim to perform its function. Whenever the patent claim tree module experiences ambiguity or failure in the processing of a claim, it will produce a log entry containing the text of the claim, the claim properties inferred by the patent claim tree module, and any corrections brought about by the user.

The patent claim tree module preferably uses the graphical display of the tree to convey visually the properties of individual claims. These properties include, but are not limited to:

* Root/Independent/ Dependent

* Claim type (Method, Composition of matter, Machine,...)

Other properties are described herein.

In some embodiments, the style of claims supported by the patent claim tree module is limited to the one prevalent in standard utility patents. As a result, the patent claim tree module covers claim sections contained in any document whose style is substantially identical to that of utility patents (e.g. Reissues, Statutory inventions). Alternative embodiments support other claim styles that are different (e.g. Design patents, Plant patents).

FIG. 87 illustrates an example patent claim tree module 8706, which is also herein called a claims processor. The patent claim tree module 8706 is composed of three main stages: A Preprocessor, an Aggregator, and a claim parser.

The responsibilities of the Preprocessor include:

* Remove any leading or trailing text
* Divide the text into a collection of claims
* Turn the input character stream into a token stream that only contains the words that may be of use to the rest of the process.

The Aggregator further turns sets of tokens following certain patterns into single tokens.

The claim parser uses the token string corresponding to each claim to set the properties of that claim as well as its dependency relationships to other claims. Its final product is preferably a directed a-cyclic graph representing the claims of the collection, their properties, and their dependencies.

An example claim tree 3702 is shown in FIG. 37. In the claim tree 3702, the root node 3704 represents the patent or the patent application. In the example of FIG. 37, the claim tree 3702 corresponds to a patent application having Attorney Docket No. 1531_0010006.

The nodes directly connected to the root node 3704, such as the nodes corresponding to claims 181, 208, 235, 271, 272, 273, and 274, represent the

independent claims in the patent or patent application. The nodes connected to these independent claim nodes represent the dependent claims of the respective independent claims. The links in the patent claim tree 3702 represent the dependencies between claims.

The invention supports a number of functions to facilitate the display and analysis of claims. For example, when the cursor is positioned proximate to or over a claim node, a portion of the claim is displayed (in other embodiments, the entire claim is displayed). In FIG. 37, for example, the cursor 3708 is displayed over node 3706 corresponding to claim 271. According to this embodiment of the invention, a portion of claim 271 is retrieved and displayed in window 3710. This portion of the claim can be retrieved from the node state information 4202, if available, or can be retrieved from a system database (such as the IPAM databases 7614 and/or the local databases 7612) by access to the link information in the node state information 4202.

The invention also supports a "display claim branch" function. As is well known, a dependent claim is construed to include the base independent claim and any intervening claims. When the user positions the cursor over a dependent claim and selects the "display claim branch" command, the system retrieves the text of the selected claim, its base independent claim, and any intervening claims between the selected claim and the base independent claim. Such retrieval can be accomplished, for example, by following the links in the claim tree 3702 from the selected claim to the associated independent claim. The invention then displays the retrieved claim language in the order of dependency. In an embodiment, the retrieved claim text is concatenated in order of the independent base claim, the intervening claims, and finally the selected dependent claim.

For example, assume that the user elected to perform a "display claim branch" function on claim 270 shown in FIG. 37 as being dependent on claim 235. The text of claim 270 is illustrated in window 4002 of FIG. 40. The text of claim 235 is illustrated in window 3902 of FIG. 39. Upon issuing the "display

claim branch" command on claim 270, the invention retrieves the text of claims 270 and 235. The invention then concatenates the retrieved text in order of dependency. The concatenated text is then displayed in a window, for example, window 4102 of FIG. 41. Note that the information displayed in window 4102 has a portion 4104 corresponding to claim 235, and a portion 4106 corresponding to claim 270.

Other display formats for presenting and displaying the retrieved claim text will be apparent to persons skilled in the relevant arts based on the discussion contained herein.

Claim trees are useful for a number of applications, including but not limited to:

- Automatically calculating claims fees for adding independent and dependent claims to a pending application or at the time a case is allowed.
- Separating out "method" and "process" claims from preamble search criteria.
- Identifying "system" claims from preamble search criteria.
- Identifying "compound" and "chemical structure" claims from preamble search criteria.
- Identifying "Jepson" type claims by searching for appropriate preamble language.
- Showing "Once Amended", "Twice Amended" and similar distinctions in claims submitted by adding these terms to the search criteria immediately following the claim number.
- Graphically showing claim dependency for ease of analysis.
- Graphically showing claim dependency to determine claim scope and coverage.
- Generating claim charts for presentations.

The claim tree functionality of the present invention supports performance of these tasks.

FIG. 38 includes a flowchart 3802 that represents the operation of the invention when generating and processing claim trees.

In step 3806, a claim is selected for processing. As noted above, the claim can be from a patent or a patent application, as long as the patent or patent application is stored in system databases.

In step 3808, the invention determines whether the selected claim is an independent or dependent claim. This is done by parsing the claim text to search for keywords that indicate whether the claim is independent or dependent. For example, the claim text may be parsed for the word "claim" in the preamble, which would indicate that the selected claim is a dependent claim.

If the claim is determined to be a dependent claim, then in step 3810 the claim text is parsed to identify the claim or claims on which it depends.

Accordingly, steps 3808 and 3810 correspond to parsing the claim language to identify claim properties and information used to generate the claim tree. Embodiments of the invention operate to recognize key words, phrases, and/or patterns in claims to identify such claim properties and information. The invention is flexible in this regard.

The need for flexibility in this area is motivated by the fact that, even though the language and structure of claims is constrained, claims are still expressed in natural language and present substantial variations in the terms and linguistic patterns used. This requires the patent claim tree module to be easily extensible to recognize new words and linguistic patterns that denote properties supported by the patent claim tree module.

Therefore, embodiments of the patent claim tree module include a configuration file. The configuration file includes information that:

* enable specification of words and phrases that denote properties identified and supported by the patent claim tree module (for example, one of these items will be the claim type of Process, which can be denoted, for example, by either the word "method" or the word "process"); and

* enable specification of words and phrases the patent claim tree module must recognize to correctly segment the text into individual claims, and the claims into sections. Two such phrases are, for example, "as claimed in claim," "in accordance with claim", or "of claim," which introduce one or more claims a claim depends on.

The configuration file can be modified.

In particular, the invention allows the addition of new textual patterns.

Also, in some cases, the user is ask to specify a textual pattern that may help the patent claim tree module deal with a given situation in the future.

Steps 3806, 3808, and 3810 are performed for each claim in the patent and/or application, as indicated by control flow line 3811. Based on the information obtained from processing steps 3806, 3808, and 3810, a claim dependency graph is generated in step 3812.

In step 3814, the claim dependency graph is mapped to a hyperbolic tree or another presentation tool.

In step 3816, the hyperbolic tree is displayed. The display of the hyperbolic tree utilizes the attributes stored in the node state information 4202 of each node. As noted above, such attributes may have been automatically established and processed based on information stored in the system, or may be user defined. The user may enter user defined attribute information by selecting an appropriate menu command, as shown in menu 4202 of FIG. 42A.

In step 3818, if the mouse is positioned proximate to or over a node, at least a portion of the text of the claim corresponding to the node is retrieved and displayed. This capability of the invention is further described above.

In step 3820, if the user selects a dependent claim and selects the "display claim branch" command (see menu 4202 in FIG. 42A), then the "display claim branch" function for the selected claim is performed. The "display claim branch" function is described above.

A user can elect to display the full text and/or image of any claim in the tree. This is done by positioning the cursor proximate to or over the node corresponding to the claim, and then selecting the "display claim" command from menu 4202.

The invention supports other processing of patent claims. For example, the invention supports automatic processing to compare a claim to an arbitrary text description.

Such processing can be used to analyze a claim for infringement purposes. In this application, the claim is compared to a description of a device or process that is accused of infringing the claim. Possible infringement is indicated if there is a close similarity between the claim and the device/process description.

Such processing can also be used to analyze a claim for patentability/validity purposes. In this application, the claim is compared to a description of one or more publications and/or events that might possibly render the claim unpatentable. Possible unpatentability/invalidity is indicated if there is a close similarity between the claim and the publications/events description.

Such processing can also be used to analyze a claim to determine if sufficient support for the claim is contained in the patent specification. In this application, the claim is compared to the specification. Probable support for the claim is indicated if there is a close similarity between the claim and the specification.

In an embodiment, the automatic claim comparison process of the invention is performed by normalizing the claim language of the claim through an implementation dependent linguistic process. The description to which the claim is to be compared is then normalized through a corresponding (sometimes identical or similar) process. The normalized texts are then compared to identify similarities and differences. A conclusion is then reached based on the context

of the analysis (i.e., whether it is for purposes of infringement, patentability, validity, support, etc.)

An example of such processing is described in U.S. Patent No. 5,754,840, which is herein incorporated by reference in its entirety.

As noted above, the patent claim tree module is capable of working in a stand-alone mode. In this mode, an example use scenario is as follows:

1. The user invokes the patent claim tree module.

2. The patent claim tree module comes up. It includes a main input/output window.

3. The user pastes a claim collection into the patent claim tree module's main input window.

4. The user configures the patent claim tree module by adjusting any available options.

5. The user press the tree generation button.

6. The patent claim tree module produces a graphical representation of the claims tree within a special purpose window. Errors and warnings are output to the log window.

The patent claim tree module can also work with IPAM. In this mode, an example use scenario is as follows:

1. The user selects a document within IPAM.

2. The user applies the "view claim tree" command to it (e.g. available inside the right-click pop-up).

3. The command routine (within IPAM) opens the document, extracts the claim section, and invokes the patent claim tree module. The claim section is passed to the patent claim tree module which comes up and pastes in its input/output window.

4. The user configures the patent claim tree module by adjusting any available options.

5. The user press the tree generation button.

6. The patent claim tree module produces a graphical representation of the claims tree within a special purpose window. Errors and warnings are output to the log window.

These use scenarios are provided for purposes of illustration only. The invention is not limited to these use scenarios. Other use scenarios will be apparent to persons skilled in the relevant art(s) based on the herein teachings.

Patent Citation Trees (patent and non-patent citations)

The invention supports patent citation trees. An example citation tree 4302 is shown in FIG. 43. In the patent citation tree 4302, each node represents a patent, and each link represents a citation. When performing a forward patent citation function, the links going from the root node toward the children nodes or leaves represent the directed association "is cited in," as in "parent node is cited in child node." When performing a backward citation function, the links going from the root node toward the leaves represent the directed association "cites," as in "parent node cites child node."

The invention supports displaying a variety of labels in the nodes of the tree. The labels may be any information pertaining to the data objects that they represent. For example, in the case where the data object is a patent, the labels may be any bibliographic information of the patent, including but not limited to the patent number, inventors, assignee, claim language (or excerpt), specification (or excerpt), drawing information such as an image of a figure, class/subclass, patent examiner, law firm, etc. The label that is displayed is user selectable.

Both patents and non-patent documents and data objects may be cited in any given patent. According to the present invention, both patents and non-patent documents/data objects are represented in the patent citation tree 4302. For example, assume that the patent citation tree 4302 is a backwards patent citation. According to this example, patent B and patent C represented by nodes

4306 and 4308, respectively, are cited in patent A represented by node 4304. Non-patent documents or data objects C and D, represented by nodes 4310 and 4312, are also cited in patent A.

According to the invention, information pertaining to the data objects in nodes of a hyperbolic tree, whether a patent citation tree or otherwise, can be retrieved and displayed by selecting the node. Such functionality is accomplished by, for example, reference to the link information in the node state information 4202, or by reference to appropriate database tables.

The display attributes of nodes can be set according to information pertaining to the data objects that they represent. In the case where the data objects are patents, for example, the display attributes of nodes can be set according to any bibliographic information of the patent, including but not limited to the patent number, inventors, assignee, claim language (or excerpt), specification (or excerpt), drawing information such as an image of a figure, class/subclass, patent examiner, law firm, etc. For example, the user can indicate that patents assigned to company X be color coded blue, whereas patents assigned to company Y be color coded red. In response to this input, the system automatically color codes nodes based on their assignee information.

Alternatively, the user can manually set the display attributes of nodes.

It is noted that this aspect of the invention has been described in the context of patent citation trees. However, the invention is applicable to data objects of any type that cite or are cited by other data objects. Accordingly, more generally, the invention supports data object citation trees.

According to embodiments of the invention, citations in citation trees are not limited to data object references that appear in any particular location of the data object(s) in question. For example, a backward citation tree for a U.S. patent is not limited to citations that appear on the front page of the U.S. patent. Also, a citation tree for a technical article is not limited to citations contained in the

bibliography of the article. Instead, citation trees can include citations to data objects that can be referenced anywhere in the data object(s) in question.

Preferably, in the case of patents, the invention includes data/text processing techniques for culling/identifying citation references to patents and non-patent documents that are contained within the specification but not listed on the first page bibliographic section. Parsing techniques to identify data in data objects are well known, and any can be used to implement this aspect of the invention. Such references can be included in patent citation trees (they could be displayed using different display attributes to denote their existence in the body of the specification, or can be displayed in different trees).

Example Databases

Databases for implementing the patent citation function of the present invention are illustrated in FIGS. 44 and 45. FIG. 44 illustrates a patent ref table 4402 that stores information on patents that were cited during the prosecution of a given patent. The patent ref table 4402 includes a record for each patent that was cited during the prosecution of a given patent application. Each record of the patent ref table 4402 includes a document_ID that identifies a base patent, and a refpatentno that identifies a reference patent (i.e., a patent that is cited in the base patent).

FIG. 45 illustrates a non-patent ref table 4404 that stores information on non-patent data objects that were cited during the prosecution of a given patent. The non-patent ref table 4404 includes a record for each non-patent data object that was cited during the prosecution of a given patent application. Each record of the non-patent ref table 4404 includes a document_ID that identifies a base patent, and a RefNo that identifies a non-patent data object that was cited in the base patent.

Information pertaining to other databases used by the invention is contained in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, and "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, incorporated herein by reference in their entireties.

The operation of the patent ref table 4402 and the non-patent ref table 4404 is illustrated in FIGS. 46 and 47. FIG. 46 presents an example patent ref table 4602 corresponding to the patent citation tree 4302 shown in FIG. 43. The patent ref table 4602 indicates that patents B and C are cited in patent A and patent D is cited in patent B.

FIG. 47 illustrates an example non-patent ref table 4702 that also corresponds to the patent citation tree 4302 in FIG. 43. The non-patent ref table 4702 indicates that non-patent data objects C and D are cited in patent A, non-patent data object A is cited in patent B, and non-patent data object D is cited in patent C.

Additional details regarding patent citation trees according to the invention are discussed in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, and "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, incorporated herein by reference in their entireties.

Data Object Family Trees

The present invention supports data object families. A data object family comprises a collection of data objects that are somehow related, and information pertaining to the relationships between the data objects in the family.

There are many types of families and family relationships. For example, the patent citation tree described above is an example data object family where the family includes a base patent and patents and non-patent data objects that are cited in or that are cited by the base patent.

According to the invention, data object families are contexts for organization, visualization, analysis, and other processing, as described above.

The invention supports the creation, generation, visualization, manipulation, and other processing of data object families. Data object families may be system defined or user defined.

Data object families are further described in the following sections.

Examples of Data Object Families

A number of the data object families supported by the present invention are described below. The invention is not limited to these examples. Additional data object families will be apparent to persons skilled in the relevant arts based on the discussion contained herein.

FIG. 51 illustrates an example patent family chronology 5102, which is a type of a data object family. A patent family chronology 5102 includes patents, applications, and related data objects that stem from a base patent application. In the example of FIG. 51, the patent family chronology 5102 stems from application 1, which represents the base patent application. Application 1 spawned a continuation application (continuation 1), which spawned a continuation-in-part application (CIP 1). A trademark (trademark 1) is related to continuation 1, and a copyright (copyright 1) is related to CIP 1. Application 1 was developed from a lab notebook maintained by an inventor (lab notebook 1). Application 1 also spawned a patent cooperative treaty application (PCT 1), which was national filed in Japan (Japan 1) and Canada (Canada 1).

FIG. 53 illustrates an example assignee technology patent family 5302, which is another type of data object family. An assignee technology patent family includes patents that are assigned to the same corporate entity, and that are technologically related. Relationships between patents may reflect, for example, improvements to the technology.

In the example of FIG. 53, the assignee technology patent family 5302 includes a bike patent 1, a bike patent 2, and a bike patent 3. Bike patent 1 is the assignee's first and base patent on bikes. Bike patent 2 is an improvement upon bike patent 1, and bike patent 3 is an improvement upon bike patent 2.

According to the invention, data object families are displayed using any well known visualization technology, including but not limited to hyperbolic trees. Techniques for generating hyperbolic trees from hierarchical organized data (such as data object families) is discussed in U.S. patent application "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, incorporated herein by reference in its entirety.

The data object families described above, as well as additional data object family examples supported by the present invention, are further described below.

Patent Related Families

The invention supports a plurality of patent-specific families. Examples of such families are described below.

Assignee Patent Family Chronology (APFC)

This family depicts the relationship between related issued patents and pending applications (and possibly related data objects) of the same assignee. This includes, but is not limited to, U.S. and corresponding foreign applications.

This includes the original application, divisions, continuations, CIPs, Reissues and Re-exams, Patent Extensions. The information to create these families could be found, for example, in the assignee's internal patent docket system.

This family would be used for understanding any company's particular family relating to a group of patents and applications that stem from the same original application. Such a family would show in a tree-type format the relationship between, for example all the patents and applications in a corporate entity's family. It helps a user see how a company has clustered improvement patents around newly developed technology. It could be useful to product planning, product development, and the business and legal people in charge of managing the company's portfolio.

Assignee Technology Patent Family (ATPF)

This family shows the relationship between patents and applications of the same assignee in the same area of technology but not necessarily limited to an Assignee Patent Family Chronology. This family would be a broader grouping that would relate patents and applications of an assignee across an entire technology. The information to create these families could be found, for example, in the assignee's internal patent docket systems and keyword searching on the assignee's patents and applications.

Technology Patent Family (TPF)

This family shows the relationship between patents in the same area of technology. This family is not limited by assignee. The information to create these families could be found, for example, by a combination of patent bibliographic and patent text searching. This family would be somewhat like a

forward citation tree but more limited based on the culling and grouping criteria used by the person creating the family.

Claim Trees

These families focus on the relationship between claims in terms of dependency and dominance-subservience among claims.

Patent Claim Tree (PCT) and Application Claim Tree (ACT)

These trees show the relationship between claims in a single issued patent or a pending application. This tree is preferably created by searching at least the first line of all claims in a given patent or application for an indication of dependency, and/or subject matter and scope. This is useful in patent prosecution, licensing and litigation scenarios in graphically representing the dependency relationship among claims in a given patent or application. Patent claim trees are further described above.

Assignee Patent Family Chronology Claim Tree (APFCCT)

This tree shows the relationship between claims in an Assignee Patent Family Chronology. This tree would show the independent-dependent claim relationships within a given patent or application and dominant and subservient or improvement claims in an Assignee Patent Family Chronology. The information to create these trees could be found, for example, in the assignee's internal patent docket systems and keyword searching on the assignee's patent

and application claim sets and the culling and grouping criteria used by the person creating the tree.

Assignee Technology Patent Family Claim Tree (ATPFCT)

This family shows the relationship between claims in an Assignee Technology Patent Family. This would be similar to the Assignee Patent Family Chronology Claim Tree but would not be limited to a given Patent Family Chronology. This family covers the Assignee's entire related technology base. This family shows the claim relationships of dependency and dominant-subservient claims within an Assignee's Technology Patent Family Chronology. The information to create these trees could be found, for example, in the assignee's internal patent docket systems and keyword searching on the assignee's patent and application claim sets and the culling and grouping criteria used by the person creating the tree.

Technology Patent Family Claim Tree (TPFCT)

This family shows the relationship between claims in all patents in a Technology Patent Family. This would be similar to the Assignee Technology Patent Family Claim Tree but would not be limited to a given Patent Family Chronology but would cover the entire related technology base for all assignees. The information to create these families could be found, for example, by a combination of patent bibliographic and patent text searching and the culling and grouping criteria used by the person creating the tree.

Patent-Trademark Relationship Trees

These families are trees that include both patent claims and trademarks used on the patented products covered by the particular patent claims. They are created in a similar way that the Patent Families and Trees would be created with the added dimension of including trademarks in the families and trees that correspond to the patents covering products sold using those trademarks.

Example Databases

Databases useful for implementing functions related to data object families are described in this section.

FIG. 48 illustrates a data object family table 4802 according to an embodiment of the present invention. Relationships between pairs of data objects are specified in the data object family table 4802. More specifically, the data object family table 4802 includes a record for each pair of data objects of interest. Each record of the data object family table 4802 includes a "from" field to identify a first data object, a "to" field to identify a second data object, and a relationship type field that stores information indicating the relationship between the two data objects.

FIGS. 50A and 50B illustrate an example relationship type table 5002 that specifies relationship types supported by the invention. The relationship types indicated in FIGS. 50A and 50B are provided for illustrative purposes, and are not limiting. In general, the relationship type table 5002 stores all relationships of interest to the user. Additional relationships will be apparent to persons skilled in the relevant art(s).

Each row of the relationship type table 5002 includes a relationship type name, the unique identifier for the relationship, and a grouping ID. The grouping ID is used to aggregate relationship types for a given data object family. For

example, grouping G1 generally corresponds to a patent family chronology. Grouping G3 generally corresponds to an assignee technology patent family.

The relationship type table 5002 includes relationships that are predefined (system defined), and relationships that are user defined. Accordingly, the invention allows users to define and/or modify relationships based on the needs of their particular applications. Additionally, the groupings contained in the relationship type table 5002 comprise both predefined groupings and user defined groupings. Accordingly, users can create and/or modify groupings based on their specific needs.

FIG. 49 illustrates an example data object family table 4902 corresponding to the patent citation tree 4302 shown in FIG. 43. The data object family table 4902 represents an alternative implementation of the patent citation tree 4302. Rows 4904-4916 all correspond to backward citation relationships, as indicated in the relationship type column. Specifically, rows 4904, 4906, 4908, and 4910 of the data object family table 4902 indicate that patent B, patent C, non-patent object C, and non-patent object D are referenced in patent A. Rows 4912 and 4914 of the data object family table indicate that patent D and non-patent data object A are referenced in patent B. Row 4916 of the data object family table 4902 indicates that non-patent data object D is referenced in patent C.

Rows 4918-4930 of the data object family table 4902 correspond to forward citation relationships, as indicated in the relationship type column. Specifically, row 4918 indicates that patent B is cited by patent A. Row 4920 indicates that patent D is cited by patent B. Row 4922 indicates that non-patent object A is cited by patent B. Row 4924 indicates that patent C is cited by patent A. Row 4926 indicates that non-patent data object D is cited by patent C. Row 4928 indicates that non-patent data object C is cited by patent A. Row 4930 indicates that non-patent data object D is cited by patent A.

Use of the data object family table is further illustrated in FIGS. 51 and 52. As discussed above, FIG. 51 illustrates an example patent family chronology 5102. FIG. 52 illustrates an example data object family table 5202 that corresponds to the patent family chronology 5102. Row 5204 of the data object family table 5202 indicates that lab notebook 1 represents invention disclosure materials used to generate application 1. Row 5206 indicates that continuation 1 is a continuation of application 1. Row 5208 indicates that CIP 1 is a continuation-in-part of continuation 1. Row 5210 indicates that copyright 1 is a copyright that is related to CIP 1. Row 5212 indicates that trademark 1 is a trademark that is related to continuation 1. Row 5214 indicates that PCT 1 is a PCT application from application 1. Row 5216 indicates that Japan 1 is a Japanese foreign national application from PCT 1. Row 5218 indicates that Canada 1 is a Canadian foreign national application from PCT 1.

In an embodiment, the patent family chronology 5102 of FIG. 51 is preferably generated by selecting grouping G1 from the relationship type table 5002. A data object family is then generated that includes the relationships within grouping G1. Further details regarding the generation of data object families is provided below.

Another example of the use of the data object family table is presented in FIGS. 53 and 54. As indicated above, FIG. 53 illustrates an example assignee technology patent family 5302. FIG. 54 illustrates a data object family table 5402 that corresponds to the assignee technology patent family 5302.

Row 5404 of the data object family table 5402 indicates that bike patent 2 is an improvement of bike patent 1. Row 5408 indicates that bike patent 3 is an improvement of bike patent 2. Additional information is contained in the data object family table 5402. For example, row 5406 indicates that bike patent 1 is technically related to bike patent 2, and is a base patent. Row 5410 indicates that the claims of bike patent 2 are subservient to the claims of bike patent 1. Row 5412 indicates that the claims of bike patent 2 are an improvement of the claims

of bike patent 1. Row 5414 indicates that the claims of bike patent 1 are dominant over the claims of bike patent 2. Rows 5410-5414 are useful for generating an assignee patent family chronology claim tree, described above.

Operational Description

FIG. 55 illustrates a flowchart 5502 that represents the operation of the invention when generating data object family tables. Depending on the particular relationships of interest, the sources of relationship data, the extent of the families of interest, and other application dependent factors, the data object family tables may be manually generated, automatically generated, or combinations thereof.

In step 5506, relationship information is obtained. Preferably, relationship information is obtained in a pair wise fashion. In other words, pairs of data objects that are somehow related are first identified. Then, the relationships between the data object pairs are determined. Alternatively, relationship information is obtained and then converted into a pair wise format.

There are a variety of sources of relationship information. Some of these sources applicable with the present invention are illustrated in FIG. 56. It is noted that the invention is not limited by the sources depicted in FIG. 56. Relationship information may be available and obtained from other sources, including but not limited to data sources identified and/or provided by users.

One source of relationship information is data having referential integrity, as indicated by 5604. Referential integrity refers to information that directly indicates a relationship between data objects. For example, the information in the patent ref table 4402 of FIG. 44 has referential integrity because the information contained therein directly establishes citation relationships between patents. Similarly, the non-patent ref table 4404 in FIG. 45 has referential

integrity because the information contained therein directly establishes citation relationships between patents and non-patent data objects.

Referring to FIG. 56, relationship information can also be obtained from relational database tables 5606. By definition, relational databases relate data objects to one another. Accordingly, the relationship represented in relational databases can be used to generate relationship tables and data object family tables.

Relationship information can also be obtained by searching database tables, as indicated by 5608 in FIG. 56. Such operations may include, for example, searching a "corporate entity" column of a database table for all occurrences of the name "Aurigin," to thereby identify all data objects that relate to the company Aurigin. This might result in identifying all patents that are assigned to Aurigin. This information is useful for generating patent family chronologies, for example.

Relationship information can also be obtained by performing text searches among data objects. Such operations may involve, for example, searching the text of documents for the word "microprocessor" to thereby identify data objects that potentially relate to the microprocessor field. This information is useful for generating technology patent families, for example. Searches could include more complex Boolean, proximity, natural language, and any types of linguistic analysis.

Additional relationship information may be obtained by performing manual acquisition and analysis of data, as indicated by 5612 in FIG. 56. Such operations may involve, for example, manually reviewing the claims of a collection of patents to determine those that are technically related, dominant and subservient relationships, base relationships, improvements, etc.

Further relationship information can be obtained by combinations of the techniques illustrated in FIG. 56.

Upon completing step 5506, data object pairs having some relationship of interest have been identified, and the relationships between such data object pairs have been determined. In step 5508, data object family tables are populated using the relationship information obtained in step 5506. For example, suppose that in the course of performing step 5506 it was determined that a particular Japanese application, called Japan 1, was filed from a particular PCT application, called PCT 1. In this case, a row in a data object family table corresponding to this pair wise relationship is created in step 5506. Such a row is shown, for example, as 5216 in FIG. 52.

Upon completing step 5508, one or more data object family tables have been created.

FIG. 57A illustrates a flowchart 5702 for analyzing relationship information according to embodiments of the invention. For illustrative purposes, and without limitation, flowchart 5702 is described with reference to the example patent family chronology 5102 shown in FIG. 51, and the corresponding data object family table 5202 shown in FIG. 52.

In step 5706, the user selects one or more data objects. The selected data objects represent the base data objects upon which the analysis is performed. In the example of FIG. 51, the user selects application 1, which is the base patent application for the patent family chronology 5102.

In step 5708, the user selects one ore more relationship types upon which the analysis is formed. In an embodiment, the user manually selects the relationship types of interest. In other embodiments, the user selects one or more grouping that contain the relationship types of interest. In the example of FIG. 51, the user selects grouping G1 (see FIGS. 50A and 50B), which contains all of the relationship types of interest for the patent family chronology (it is noted that grouping G1 can be modified as desired by the user to perform a custom patent family chronology analysis).

In step 5710, the user indicates whether complete or partial closure is required. If the user selects complete closure, then all relationships stemming from the data objects selected in step 5706 are identified. If the user selects partial closure, then only a portion of the relationships stemming from the selected data objects are identified (for example, perhaps only relationships moving forward or moving backward from the selected data objects are identified). Also in step 5710, if the user selects partial closure, then the user may be required to provide information regarding the scope of the desired analysis.

In step 5712, closure based on the selected data objects, the selected relationship types, and the scope of closure is computed. In other words, all relationships from the data object family tables that satisfy the information and commands provided in steps 5706, 5708, and 5710 are identified.

Preferably, step 5712 is performed as shown in the flowchart of FIG. 57B.

In step 5718, rows in the data object family tables that match the selected data objects and the selected relationship types are identified. A row matches a selected data object if either the "from" field or the "to" field of the row contains the selected data object. A row matches a selected relationship type if the relationship type field of the row contains one of the selected relationship types.

Step 5720 is optionally performed depending on the level of closure specified in step 5710. In step 5720, additional rows in the data object family tables that match data objects specified in rows identified in step 5718, and that also match one or more of the selected relationship types are identified. For example, in the data object table 5202 of FIG. 52, row 5206 is identified in step 5718 because application 1 (the data object selected in step 5706) is contained in the "from" field. Note that continuation 1 is contained in the "to" field. In step 5720, rows in the data object family tables that match continuation 1, and that also match one or more of the selected relationship types are identified. Thus,

row 5212 is identified because continuation 1 from row 5206 is contained in the "from" field, and also the trademark relationship (which is part of selected group G1) is contained in the relationship type field.

Steps 5718 and 5720 are repeatedly performed until the level of closure specified in step 5710 is satisfied, as indicated by control flow line 5721.

Upon the completion of steps 5718 and 5720, data object pairs having relationships of interest have been identified. Also, the relationships between such data object pairs are identified. In step 5722, a tree of the data object pairs is generated. Preferably, the tree is generated by selecting a data object pair, adding the selected data object pair and its associate relationship to the tree, and then selecting a new data object pair. The process continues until each data object pair has been added to the tree.

For example, assume that steps 5718 and 5720 identified rows 5204-5218 in FIG. 52. In step 5722, the tree or graph is generated from rows 5204-5218. For example, processing of row 5204 results in creating nodes 5106 and 5108 and link 5104 in the tree (see FIG. 51). Processing of row 5206 results in adding node 5110 and link 5112 to the tree (FIG. 51). The other identified rows 5208-5218 are processed in a similar manner in step 5722.

Referring again to FIG. 57A, in step 5714, the data object family is displayed. According to the invention, the tree generated in step 5722 (FIG. 57B) may be displayed using any visualization technology, including hyperbolic trees. Techniques for generating hyperbolic trees are discussed in U.S. patent application "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, incorporated herein by reference in its entirety.

Plug-In Architecture

According to the invention, the enterprise server 7604 (FIG. 76) can interface with third party tools to enhance and extend its functionality. As used herein, the phrase "third party tools" refers to any hardware, software, or combination thereof that is external to the enterprise server 7604 or any of its component or associated modules, such as the web server 7610, the databases 7612 and 7614, the network client 7608, and the web client 7606. The third party tools are selected and integrated with the enterprise server 7604 on the basis of their ability to perform desired functionality not available internally (i.e., not performed by the enterprise server 7604 or any of its components or associated modules). Such functionality can pertain to data acquisition or storage, visualization, display, processing, and/or any other data processing.

FIG. 59 illustrates a flowchart 5902 representing illustrative operation of the enterprise server 7604 when interacting with a third party component.

In step 5906, a user begins analyzing a collection of data objects. The data objects may be, for example, from one or more groups, and/or one or more data object types. Alternatively, the data objects may have resulted at least in part from processing performed by the enterprise server 7604, such as from conducting database searches.

In step 5908, the user analyzes the data objects using functions provided by the enterprise server 7604. Such functions are described above. Additional functions are described in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, incorporated herein by reference in their entireties.

In step 5910, the user wishes to run one or more reports based on the analysis performed in step 5908. Some of the reports may be generated using

functions provided by the enterprise server 7604. Such reporting functions of the enterprise server 7604 are described herein, and are further described in the patent applications referenced above. Other reports may be generated using third party components that interface with the enterprise server 7604.

In step 5912, the user wishes to further analyze the data objects using functions provided by third party components that interface with the enterprise server 7604. Information pertaining to the data objects are sent to the third party components, along with commands, options, and other information needed to allow the third party components to perform the analysis requested by the user. The third party components perform the requested functions using the information received from the enterprise server 7604.

In step 5914, the user instructs the third party components to send the results of their respective processing back to the enterprise server 7604. Such results may be stored and utilized as a new group or data object type in the enterprise server, for example. Such results could also be processed alone or with other data, and/or could be stored as data in fields of data records and/or reports.

FIG. 64 is a flowchart 6402 further depicting the operation of the enterprise server 7604 when interacting with a third party component according to an example embodiment. FIG. 63 is an example data flow diagram used to explain the operation of the flowchart 6402 in FIG. 64.

In step 6406, the user wishes to analyze a group 6304 containing one thousand documents or other data objects. The documents include both patents and non-patent documents. The documents relate to computer programming languages.

The user in step 6406 selects a third party component to organize and visualize the content of the documents. The display generated by the selected third party component is shown as 6306 in FIG. 63. In display 6306, documents that are similar in content are represented as peaks in a landscape. The relative

size of the peaks correspond to the number of documents that are directed to the subject matter associated with the peaks. Preferably, operation of the third party component is dynamic, such that the context of the landscape is dynamic. For example, the display 6306 shown in the example of FIG. 63 is based on the subject matter or topic of the documents in the group 6304. Preferably, the user can access a menu 6308 which allows the user to change the context of the display from topical to another attribute, such as author. If the user changes the context to author, then the peaks in the display 6306 would correspond to author, and the relative size of the peaks would indicate the number of documents that were by respective authors.

In step 6408, the user selects a subset of the 1,000 documents via use of the third party component. In the example of FIG. 63, the user has selected from the display 6306 the peaks corresponding to PASCAL and COBAL. This selection is indicated by 6307.

In step 6410, the user instructs the third party component to send the results of the analysis back to the enterprise server 7604. In the example of FIG. 63, the third party component sends back an indication of the selection 6307. For example, the third party component may send back a list of the documents corresponding to the selection 6307 (that is, a list of the books directed to PASCAL and COBAL).

In step 6412, the enterprise server 7604 creates a new group from the information received from the third party component. In this case, the new group contains the documents corresponding to the selection 6307 that pertain to PASCAL and COBAL.

In step 6414, the enterprise server generates and saves a search query corresponding to the analysis performed by the user using the third party component. In example of FIG. 63, the search query generated and saved by the enterprise server 7604 comprises a search for documents pertaining to PASCAL

and COBAL. The search query is generated and saved so that it can be referred to and used in the future by the user.

The new group 6310 can then be used in the enterprise server 7604 just like any other group. Specifically, processing can be performed on the documents in the new group 6310 as described above, and as further described in U.S. patent applications "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, "System, Method and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, herein incorporated by reference in their entireties.

The examples discussed above illustrate the round trip functionality of the present invention. Specifically, the invention allows information to be transferred from the enterprise server 7604 to third party components. Also, the invention enables the transfer of information from third party components to the enterprise server 7604. In certain instances, depending on the functionality desired and the third party component being used, only one way interaction is involved. For example, often when working with a report generator, data flows only from the enterprise server 7604 to the report generator. Such data is sent to the report generator to enable the report generator to create a desired report. With some report generators, data does not flow back from the report generator to the enterprise server 7604 (although, with some report generators, data may flow back to the enterprise server 7604. For example, the report generator may send an electronic copy of the report back to the enterprise server 7604).

Plug-In Architecture

FIG. 58 is a block diagram 5802 of the enterprise server 7604 when interacting with third party components/tools 5808. As indicated in FIG. 58, the enterprise server 7604 interacts with third party components/tools 5808 via an interface 5806.

FIG. 60 illustrates a more detailed block diagram of the enterprise server 7604 when interacting with third party components/tools 5808.

The enterprise server 7604 includes an enterprise server API (application programming interface) 6004. Generally, the enterprise server API 6004 includes a collection of commands that the enterprise server 7604 understands. These commands instruct the enterprise server 7604 to perform specific functions. An entity external to the enterprise server 7604 interacts with the enterprise server 7604 by sending commands that conform to the enterprise server API 6004 to the enterprise server 7604. The enterprise server API 6004 is further described below. The enterprise server API 6004 is additionally described in U.S. patent application U.S. Patent Application "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, incorporated herein by reference in its entirety.

The interface 5806 is preferably represented by a plug-in manager 6006. The plug-in manager 6006 includes a plug-in manager API 6008. The plug-in manager API includes commands which the plug-in manager 6006 understands. These commands instruct the plug-in manager 6006 to perform specific functions. Entities external to the plug-in manager 6006 interact with the plug-in manager 6006 by sending commands that conform to the plug-in manager API 6008 to the plug-in manager 6006.

The third party components/tools 5808 are represented by plug-ins 6012 and third party components/tools 6024. The third party components/tools 6024 represent any hardware, software, or combination thereof that performs desired functionality. The third party components/tools 6024 each includes an API.

Devices external to the third party components/tools 6024 interact with the third party components/tools 6024 via their respective APIs.

Preferably, a plug-in 6012 exists for each third party component/tool 6024. Alternatively, one or more third party components/tools 6024 may share a common plug-in 6012.

Generally, the plug-ins 6012 represent translation modules. Messages sent from the plug-in manager 6006 to a third party component/tool 6024 is translated from a format native to the plug-in manager 6006 to a format recognized by the API of the third party component/tool 6024. Similarly, a message that is send from a third party component/tool 6024 to the plug-in manager 6006 is translated by the plug-in 6012 from the native language of the third party component/tool 6024 to commands and/or messages that conform to the plug-in manager API 6008.

As indicated in FIG. 60, the enterprise server 7604, the plug-in manager 6006, the plug-ins 6012, and the third party component/tools 6024 interact with a database 6010, which may include, for example, the IPAM databases 7614.

The process of translating between the plug-in manager API 6008 and the API of any given third party component/tool 6024 will be apparent to persons skilled in the art based on the discussion of the plug-in manager API 6008 contained herein, and knowledge of the well known API of the third party component/tool 6024. Additionally, interaction with a third party component/tool 6024 via its API will be apparent to persons skilled in the relevant arts.

The plug-in architecture described herein is applicable to both the server and the client. In embodiments of the invention, the client plug-in architecture works in a manner substantially similar to the server plug-in architecture, a difference being in the type and kind of APIs supported/required by the client and the manner in which the client detects that plug-ins are present. Implementation

of the server and client plug-in architectures will be apparent to persons skilled in the relevant art(s) based on the discussion contained herein.

Plug-In Operation

FIGS. 62A and 62B collectively illustrate a flowchart 6202 representing the operation of the plug-in manager 6006 when interacting with a third party component 6024 according to an embodiment of the invention. FIG. 61 illustrates an example event trace diagram corresponding to the flowchart of FIG. 62A and 62B.

In step 6206, the enterprise server 7604 creates an instance of the plug-in manager 6006. This is indicated as 6120 in FIG. 61.

In step 6208, the plug-in manager determines which plug-ins 6012, if any, are available. This is indicated as 6122 in FIG. 61. As noted above, each plug-in preferably corresponds to a third party component 6024, and understands the API, the information requirements, the options, etc., of the corresponding third party component 6024 (i.e., how to interact with the third party component).

In step 6210, the enterprise server receives from the plug-in manager 6006 a list of the available plug-ins 6012. This is indicated as 6122 in FIG. 61. This list indicates which third party components 6024 are available for use by the user.

In step 6212, the user pops up a plug-in menu 6104. The menu 6104 lists all plug-ins 6012 that are available based on the list obtained in step 6210. Further in step 6212, the user selects a plug-in from the plug-in menu 6104.

In step 6214, the enterprise server 7604 instructs the plug-in manager 6006 to run the selected plug-in. This is indicated by 6124 in FIG. 61.

In step 6218, the plug-in manager 6006 instructs the selected plug-in 6012 to run. This is indicated by 6128 in FIG. 61.

In step 6220, the plug-in 6012, possibly with the assistance of the plug-in manager 6006, displays an options dialog or a series of dialogs to the user. The dialogs specify the functions that can be performed by the corresponding third party component 6024. Preferably, at least all functions that are accessible through the third party component's API are listed in the dialogs. In other words, interaction between the enterprise server 7604 and the third party component 6024 includes at least the functions and capabilities represented in the API of the third party component 6024. The user selects desired options via the displayed options dialog. This interaction is represented by 6130 in FIG. 61.

In step 6222, the plug-in 6012 makes requests for data from the plug-in manager 6006. This is represented by 6138 in FIG. 61. In response to such requests, the plug-in manager 6006 may make calls to the enterprise server 7604 (represented by 6134 in FIG. 61), or may directly access the databases 6010 (this is represented by 6136 in FIG. 61).

In step 6224, the plug-in 6012 creates an instance of the associated third party component 6024. This instance is a software representation of the third party component. For example, if the third party component is a report generator, then in step 6224 the plug-in invokes the report generator. This is represented by 6140 in FIG. 61.

In step 6226, the plug-in transfers data to the third party component 6024. The plug-in performs any necessary data transformations. Additionally, the plug-in 6012 sends commands and options (from step 6220) to the third party component 6024, or alternatively controls the third party component 6024 based on the commands and options from step 6220. Preferably, the plug-in 6012 interacts with the third party component 6024 in this manner using the API of the third party component 6024. This operation of step 6226 is represented by 6142 in FIG. 61.

Also in step 6226, the user performs any real time interaction with the third party component 6024 via the plug-in manager 6006 and the plug-in 6012.

An example of such interaction was described above with reference to FIGS. 63 and 64.

In step 6228, the user instructs the third party component 6024 to export the results of any processing performed by the third party component 6024. This is represented by 6144 in FIG. 61.

In step 6230, in response to the export command, the third party component 6024 transmits data to the enterprise server 7604 via the plug-in 6012 and the plug-in manager 6006. The plug-in 6012 and/or the plug-in manager 6006 perform any necessary data transformations. The enterprise server 7604 receives and processes the information as described above.

Interaction with third party components shall now be considered in greater detail.

Embodiments of the invention support a number of methods for integration with third party components. Two such integration methods are called "loose integration" and "tight integration." These are described below.

Loose Integration

With loose integration, interaction is via an intermediary, such as files. To export data from application 1 (such as IPAM) to application 2 (such as a third party tool), application 1 or an entity working on behalf of application 1 stores the data in a file (preferably in a format understandable by application 2; otherwise, data translations may be performed). The file is then read by application 2, or an entity working on behalf of application 2.

To import data to application 1, application 2 or an entity working on behalf of application 2 saves data in a file. The file is then read by application 1, or some entity working on behalf of application 1 (some data translation may be necessary). In embodiments of the invention, these write and read operations are

manually initiated. That is, the user has to manually initiate the exportation and importation of data.

Generally, it is possible to utilize off-the-shelf versions of a third party component when loose integration is used.

Tight Integration

With tight integration, IPAM and a third party component interact with each other via the plug-in architecture described herein. Data is sent from IPAM to a third party component via the plug-in mechanism, and visa versa.

In an embodiment, IPAM (via the plug-in mechanism) sends a list of pointers to the third party component. The pointers point to data (i.e., data objects). In this manner, IPAM sends data to third party components. This same process is also used to send data from third party components to IPAM (via the plug-in mechanism). Alternatively, IPAM could send data to a third party component by storing the data in a file, and then sending a message to the third party component (via the plug-in mechanism) with the name of the file. This process could also be used to send data from third party components to IPAM. Alternatively, IPAM could send data to a third party component by sending messages (containing the data) via the plug-in mechanism to the third party components. This process could also be used to send data from third party components to IPAM.

Tight integration is a more automated process. The user need not explicitly store data in files, or manually initiate read and/or write operations. Instead, the user need only issue import or export commands. An example of the user issuing an export command is shown, for example, in FIG. 90. In the example of FIG. 90, the user is exporting the data objects in a group "adewolfe" to Microsoft Excel. The menu 9002 shown in FIG. 90 also illustrates other commands 9004 to import and export.

In order to perform tight integration, it is necessary that the API of the third party component include sufficient functionality to support the interactions described above. Alternatively, tight integration can be achieved using a version of a third party component that has been modified to support the interactions described above.

Query Creation and Retention

According to the invention, upon receipt of information from a third party component, the enterprise server 7604 creates a search query that corresponds to the analysis/processing performed by the user using the third party component. In an embodiment, the enterprise server 7604 generates the query by comparing the information sent to the third party component with the information received from the third party component.

Consider the example of FIG. 63, where 1000 documents pertaining to computer programming languages were sent to the third party component, and a subset of these documents pertaining to Pascal and Cobol were returned to the enterprise server 7604. According to an embodiment, the enterprise server 7604 compares what was sent with what was received to generate a search query limited to Pascal and Cobol.

In other embodiments, the third party component returns status information that the enterprise server 7604 uses to generate the search query. In other embodiments, the third party component returns the search query.

The query is saved for later reference and use by the user.

Plug-in Manager Application Program Interfaces (APIs)

The Plug-in Manager is preferably an ActiveX component that supports the addition of "plug-ins" (also preferably ActiveX components) that support

interaction with third-party components. As noted above, plug-ins are used to export data to and import data from third-party visualization components (such as but not limited to Microsoft Excel, Cartia ThemeScape, and SpotFire).

The Plug-in Manager defines several APIs, described below, for use by various parts of the Plug-in facility. The Plug-in Manager also provides a local database, which provides "linked tables" (proxies) to tables in the IPAM SQL Server database.

When the IPAM is started, the IPAM queries the Plug-in manager for available Plug-ins, preferably creating a pull-right menu listing their names. (If no Plug-ins are available, the menu is not created, and access to Plug-ins is disabled in the IPAM.) In one embodiment, a user selects an IPAM group, then uses the Plug-in menu to cause the Plug-in Manager to invoke a specific Plug-in. Preferred protocols by which this happens are described below.

Additional APIs applicable with the invention are described in "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 07/867,392, filed June 2, 1997, which is incorporated herein by reference in its entirety. Implementation of the APIs and commands described below will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

It should be understood that the following APIs and commands are provided for illustrative purposes only, and are not limiting. Any other APIs or commands that achieve the functions and requirements described herein can also be used, as will be apparent to persons skilled in the relevant art(s) based on the herein teachings.

API: IPAM to Plug-in Manager

The Plug-in Manager exposes an API to the IPAM. This API allows the IPAM to obtain the number and names of available Plug-ins, and to invoke those Plug-ins, but preferably shields the IPAM from Plug-in-specific details.

In an example embodiment, plug-ins advertise their availability by way of a Windows Registry subkey under the key:

HKEY_LOCAL_MACHINE\Software\Aurigin Systems, Inc.\Aurigin Client\Plug-ins\

The name of the subkey is the OLE class name of the Plug-in (e.g., Aurigin.ExcelPlug-in).

If the default value of the subkey is "0", then the Plug-in is considered to be unavailable. Any other value indicates that the Plug-in is available. In other embodiments, the Plug-in Manager may automatically mark a Plug-in as being unavailable under certain circumstances. (For example, if a Plug-in fails some set number of times in succession).

The "Name" value of the subkey holds the name of the Plug-in, as it is to appear in IPAM menus.

The "Description" value of the subkey holds a string that provides additional information about the Plug-in. For example, to be used as a tooltip.

Various example commands/functions are described below.

Plug-inCount() As Integer

Returns the number of available Plug-ins.

The Plug-inManager queries the Registry to identify available Plug-ins, as described above.

Plug-inName(index As Integer)

Returns the name of the index'th Plug-in, where 0 <= index < Plug-inCount().

Plug-inDescription(index As Integer) As String

Returns the description of the index'th Plug-in, where 0 <= index < Plug-inCount().

SetGroup(name, id As String)

Sets the name and database id of the group that will be the source of data for a Plug-in.

As a side-effect, the list of subgroups and the list of documents is cleared.

AddDocument(name, id As String)

Adds the name and id of a document to the list of documents in the group. After the IPAM invokes SetGroup(), the IPAM is required to invoke AddDocument() once for each document in the group.

SetAttribute(name, value As String)

Sets or resets a dictionary entry for "name" to be "value". The dictionary can be queried by a Plug-in using protocol described further below. The IPAM is required to set the following attributes

UserId	the ID of the logged-in user
IPAMServer	the name and port number of the IPAM server (e.g., "Romulus:8080")

SQLServer the name of the server on which SQL Server is running

RunPlug-in(Integer index)

Run the index'th Plug-in, where 0 <= index < Plug-inCount().

The Plug-in manager creates an instance of the ActiveX component for the Plug-in, and invokes it using the "Plug-in Manager to Plug-in" API, described below.

If the Plug-in "fails" (where conditions constituting failure are defined by the specific Plug-in), the Plug-in will raise an OLE exception. If no exception is raised, the Plug-in is assumed to have succeeded.

API: Plug-in Manager to Plug-in

Each Plug-in preferably implements this API, through which the Plug-in Manager invokes (runs) the Plug-in.

Various example commands/functions are described below.

Run(Object manager)

Run the Plug-in. The "manager" object is a reference to the instance of the Plug-in Manager ActiveX component. The Plug-in uses the "Plug-in to Plug-in Manager" API, described below, to communicate with the Plug-in manager.

When run, a Plug-in preferably will do one or more of:

* present a dialog to the user listing Plug-in-specific options
* query the Plug-in Manager for the selected group and its documents

* direct the Plug-in Manager to perform citation traces for documents
* query a local database for additional information about documents
* invoke a third-party API to communicate with a third-party application or component (e.g., Microsoft Excel, Cartia ThemePublisher, SpotFire, Inxight Hyperbolic Tree)
* insert new data into a local database
* invoke the IPAM "Plug-in to IPAM" API to create new groups of documents
* present one or more status of informational dialogs to the user

If the Plug-in encounters an error, it communicates details back to the Plug-in Manager by raising an OLE exception, passing a description of the error in the exception.

API: Plug-in to Plug-in Manager

The Plug-in Manager implements this API for use by Plug-ins. Various example commands/functions are described below.

GetAttribute(name As String) As String

Get the value of an attribute, as set by SetAttribute(). The Plug-in uses this function as necessary to obtain the id of the logged-in user, the name of the server name and port number of the IPAM server, etc.

Raises an exception if the name is unknown.

UnrollGroups(depth As Integer)

IPAM Groups form a hierarchy. UnrollGroup populates the bds_groups table in the local database with a trace of the group hierarchy from the source group for the given depth. (A depth of 0 means the source group only, 1 means to include the source group and one level of subgroups, and so on.)

The Plug-in then gains access through the GetDB() function, described below, and can issue SQL queries that join against the bds_groups table.

GroupCount() As Integer

Returns the number of source groups. In an embodiment, this always returns 1.

GetGroups(name() As String, String id() As String)

Gets the names and ids of groups.

The Plug-in is responsible for sizing the name and id arrays to accommodate GroupCount() groups.

DocumentCount() As Integer

Returns the number of documents in the source group(s).

GetDocuments(name() As String, id() As String)

Gets the (long display) names and ids of documents in the source group(s).

The Plug-in is responsible for sizing the name and id arrays to accommodate DocumentCount() documents.

TraceCitations(forward As Boolean, depth As Integer)

Directs the Plug-in manager to trace citations for the source documents, writing the results into the 'bds_citations' table in the local database. Citations are traced either forward or backward, and to a given depth.

The Plug-in then gains access to the local database via GetDB(), as described below, and can issue SQL queries that join against the bds_citations table.

GetDB() As Database

Return an ActiveX object that provides access to the local database. "Database" is an OLE Object type provided by Microsoft. It provides an API for interacting with JET databases. The database object has an API that supports SQL queries.

Through the local database, the Plug-in gains access to tables in the IPAM database, and optionally to third-party or customer-specific tables, allowing information to be merged for presentation.

API: Plug-in to IPAM

The IPAM exposes an API for creating new groups.

Various example commands/functions are described below.

CreateGroupFromGUIDs(name, description As String, guid() As String)

Create a new group with the given name and description, populating it with documents.

Preferably, use of this API requires that:

1. a user be logged in to the IPAM

2. a group is selected in the group pane

3. the user has permission to create a new subgroup within the selected group

If the guid() array contains ids that are not recognized, those ids are written to a log file.

Example Third Party Components

The enterprise server 7604 can interact with any third party component having functionality and capabilities of interest. As indicated above, the enterprise server 7604 interacts with a given third party component 6024 via the API of the third party component 6024. Alternatively, the enterprise server 7604 interacts with a third party component 6024 via a custom interface to the third party component 6024.

For illustrative purposes, and without limitation, example third party components 6024 are described below. It should be understood that the present invention is not limited to working with these third party components. The following discussion is provided solely to illustrate example operation of the enterprise server 7604 with third party components 6024, and is not limiting.

Graph-Based Visualization Tools

FIG. 67 illustrates an example display where data is represented in a graph. The example of FIG. 67 depicts a two-dimensional graph, but graphs of dimensions greater than two are within the scope and spirit of the invention.

In the example display 6702 of FIG. 67, each icon in the graph 6708 represents one or more patents. In an embodiment, information pertaining to a patent (such a bibliographic information, text information, image information,

etc.) can be retrieved and displayed by selecting the icon associated with the patent.

The X-axis of the graph represents the issue date, and the Y-axis of the graph represents the filing date. The patent icons are positioned on the graph 6708 depending on their particular filing dates and issue dates. The display can be dynamically adjusted by selecting other criteria for the X-axis via a drop down menu 6710. Similarly, the display can be dynamically adjusted by selecting other criteria for the Y-axis via a drop down menu 6712.

The data represented in the graph 6708 can be filtered by selecting from attributes 6706. In example of FIG. 67, for example, the patents represented in the graph 6708 are limited to art units 111 through 358, and remaining pendency of 42 weeks to 961 weeks.

A variety of graph based visualization tools operable with the present invention are publicly available. For example, and without limitation, SPOTFIRE PRO is a commercial product available from SPOTFIRE, Boston, Massachusetts. SPOTFIRE PRO enables the display of data in graphical form as shown in FIG. 67.

Landscape-Based Visualization Tools

FIG. 65 illustrates an example display 6502 wherein data is represented in a landscape or map. Peaks in the landscape correspond to data objects that have been aggregated according to user selected criteria. In the example of FIG. 65, the peaks correspond to claims from one or more patents that are directed to similar subject matter. For example, reference number 6508 identifies a peak that represents an aggregation of claims from one or more patents that are directed or related to "policy". The relative heights/size of the peaks indicate the number of claims corresponding to the respective subject matters of the peaks.

The display can be dynamically adjusted by selecting attributes from a menu 6510. For example, if the "assignee" attribute is selected, then the landscape would be adjusted so that the peaks would represent the number of patents assigned to corporate entities.

Information in the display can be limited by using filters from a filter menu 6512. For example, if an "assignee" attribute is selected, and "Aurigin" is entered in a pop-up dialog, then the landscape would be adjusted to only display patents assigned to "Aurigin."

According to the example shown in FIG. 65, when the cursor 6504 is positioned proximate to or above a peak, the text of a claim that corresponds to the peak is partially or fully displayed. This is illustrated by window 6506.

FIG. 66 illustrates a similar display 6604 that represents patent related data using a landscape approach.

FIG. 63, described above, is directed to another display 6306 wherein information is visualized using a landscape approach.

As described above, and further illustrated in FIG. 88, information from IPAM 8806 can be exported to the landscape tool 8808. This is represented by data flow line 8802. The exported information is represented as a map 8810. The user can manipulate the map to select groups of data objects, which can then be imported to IPAM 8806. This is represented by data flow lines 8804. The imported data can be used to form new groups, as shown in FIG. 88.

An example product that utilizes a landscape approach to visualize data is THEMESCAPE from CARTIA, INC., Bellevue, Washington. FIG. 68 illustrates an event diagram 6802 representing the interaction between the enterprise server 7604, the enterprise client 7606 or 7612, and the third party THEMESCAPE product 6807 according to an embodiment of the invention. The invention is not limited to this example implementation.

Interaction with Cartia ThemeScape will be further illustrated by reference to an example scenario shown in FIGS. 91-98. This also further illustrates the round-trip capabilities of the invention.

First, an IPAM group is selected, and this is sent to Cartia ThemeScape. See FIG. 91.

Second, available options are chosen in the Cartia Plugin dialog (see FIG. 92). The dialog allows you to change the name of the map (the name of the IPAM Group is the default map name). The dialog also allows you to specify what portions of the patent documents in the group to use to build the map. Dialogs vary among third party tools.

Third, a notification is provided indicating that Cartia has begin processing the map. See FIG. 93.

Fourth, progress is indicated in the ThemePublisher's Map Manager dialog. See FIG. 94. When it has built the map, the status line will say "Processing Complete", or will indicate why the map could not be built.

Fifth, the map is displayed. See FIG. 95. The user uses ThemePublisher's tools to select a portion of the map.

Sixth, with a portion of the map selected, the ThemePublisher Document Viewer appears. See FIG. 96. The user clicks the Export button to export the data objects corresponding to the selected map portion to IPAM.

Seventh, an Export dialog is presented. See FIG. 96. In the example of FIG. 96, the "Aurigin IPAM Group" option is selected, and then the user clicks OK. In some embodiments, Cartia ThemePublisher displays a dialog that asks, "What should the Aurigin IPAM group be named?" with a field for the user to enter a name. It will contain a default value such as the map name.

Eighth, a new group is created as the child of whatever group is selected.

This example has been provided for purposes of illustration, not limitation. The example is applicable to other third party tools. Accordingly, given the herein teachings, persons skilled in the relevant art(s) could integrate

other third party tools with IPAM. The invention is directed to such other combinations.

Data Object Searching Tools

The invention is operable with third party tools for identifying collections of data objects. Such data objects can be imported into IPAM for processing. For example, one or more new groups may be formed from the imported data objects.

For example, FIG. 89 illustrates a chemical search engine 8904. (The chemical search engine 8904 is herein described for illustrative purposes only. The invention is not limited to this embodiment.) The user defines a chemical structure 8905 in the chemical search engine 8904. The chemical search engine 8904 then searches for chemicals that are related to the chemical structure 8905. A variety of chemical search engines are publicly available and operable with the invention, such as CAS SciFinder.

The results of the search are stored via a save dialog 8906. In the example of FIG. 89, the search results are saved in file "test.csv." The search results comprise one or more chemicals that were found during the search. Generally, each chemical represents a data object.

In IPAM 8902, the user imports the file "test.csv," as indicated by arrow 8908. The contents of "test.csv" (i.e., the chemicals located during the search) are stored in a group 8910.

An example use scenario is as follows.

A user working in IPAM 8902 is studying a group comprising a multitude of chemical compounds (each compound may be represented by a corporate document, for example). The user wishes to identify chemical compounds in this group that have a particular sub-structure.

The user instructs IPAM 8902 to invoke the chemical search engine 8904, and export the group to the chemical search engine 8904. Accordingly, the plug-in manager interacts with the plug-in for the chemical search engine 8904 to (1) invoke the chemical search engine 8904, and (2) export the group to the chemical search engine 8904. The plug-in interacts with the chemical search engine 8904 using the API of the chemical search engine 8904. The plug-in performs any necessary data/format translations to export the group to the chemical search engine 8904.

The user interacts with the chemical search engine 8904 to conduct a search for compounds related to a particular structure 8905, as described above. This search is limited to the compounds in the group that was exported to the chemical search engine 8904.

After completion of the search, the plug-in associated with the chemical search engine 8904 accesses the search results (possibly from file "test.csv") and imports the search results to IPAM 8902 via the plug-in manager. In doing so, the plug-in performs any needed data/format translations consistent with the API of the plug-in manager.

IPAM 8902, upon receiving the search results, may create a new sub-group of the original group, and store the search results in the new sub-group. Other processing by IPAM 8902 is possible, as will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

Other

Additional third party components applicable for use with the enterprise server 7604 include, but are not limited to, CRYSTAL REPORTS from SEAGATE, EXCEL from MICROSOFT, TABLE LENS from INXIGHT, and PERSPECTA from PERSPECTA, INC.

Hybrid Models

The invention supports a variety of configurations of its components. Additionally, the invention supports a variety of functional configurations and distributions.

For example, FIG. 69 illustrates a configuration where all primary components and functions are positioned at a single location 6904.

FIG. 70 illustrates a configuration where components and functionality are distributed among a first location 7002 and a second location 7004. The enterprise server 7604, the client 7606, 7608, and the IPAM databases 7614, and functions performed by these components, are positioned at the first location 7002. The search engines and indexes 6902, and the functions performed by the search engines and indexes 6902, are positioned at a second location 7004. To perform a search specified by the user, the enterprise server 7604 accesses the remote search engines and indexes 6902 at the second location 7004. It is noted that such search engines and indexes 6902 may include third party components, such as LEXIS-NEXIS, Westlaw, Dialog, Derwent, etc., as described above.

FIG. 71 illustrates a configuration where components and functionality are distributed among a first location 7102, a second location 7104, and a third location 7106.

FIG. 72 illustrates a configuration where components and functionality are distributed among a first location 7202, a second location 7204, a third location 7206, and a fourth location 7208.

As indicated by the example configurations illustrated in FIGS. 69-72, the present invention includes a variety of configurations and distributions of its components and functionality. The examples of FIG. 69-72 are presented for purposes of example only and are not limiting. Other configurations of components and functionality will be apparent to persons skilled in the relevant arts based on the discussion contained herein.

For example, a given component and/or functionality can be distributed among multiple locations. This is illustrated in FIG. 73, where the enterprise server 7604 is distributed among a first location 7302 and a second location 7304. Additionally, databases 7206 are distributed among location 7306 and 7308. Also, search engines and indexes 6902 are distributed among locations 7310 and 7312. Third party components/tools (described above) may also be local or distributed.

It is noted that the locations described in FIGS. 69-73 may be owned by and/or under the control of a single corporate entity, or owned and/or under the control of multiple corporate entities. Also, the locations depicted in FIGS. 69-73 may be local to one another, or may be very remote from one another, or may be combinations thereof.

The invention supports a number of economic models for using the system. For example, the system supports a pay-for-use service, where usage of the system is metered and billed accordingly. The invention also includes other economic models, such as but not limited to an unlimited service plan, where for a given fee the user can download any number of documents and have unlimited use of the system, and a hybrid plan where for a fee the user can download a given number of documents and use the system for a given amount of time or transactions (or some other metric). Once these limits are exceeded, the user pays on a per-use rate.

The invention also supports a model whereby meta-data version (v1) can be distributed to the client at time (t1). Later, an updated version of the meta-data (v2) can be distributed at time (t2). This is extremely valuable because certain aspects of the meta-data associated with a patent are mutable (e.g., assignee and U.S. classification). By providing these version numbers in the database, the host system can automatically provide meta-data updates to the remote system. Charges for these updates can be included within the user's existing economic model (above), or via a separate economic model.

The invention also supports electronic ordering of data objects (such as patents), and automatic downloading and installing of such data objects. In this embodiment, a user can electronically place an order for data objects, such as but not limited to via modem or over the Internet. An automated process takes the order, performs any necessary financial accounting and billing, retrieves the requested data items from a database, and downloads the retrieved data items/objects to the requestor via the Internet, for example. At the user computer, the received data objects are automatically decrypted (if necessary), unpacked (if necessary), unzipped (if necessary), installed in the proper directories, noted in the computer registry, and/or other tasks to automatically install the received data objects for use at the user site.

Assignee Processing

The invention supports processing related to corporate entities. For example, the invention includes functionality to identify and process patents assigned to a particular corporate entity.

To achieve effective corporate entity related processing, it is important that references to a given corporate entity in a data set be consistent throughout the data set. For example, if the user wishes to identify all patents assigned to Aurigin, Inc., then it is important that all references to Aurigin in the database utilize the same name, such as "Aurigin, Inc.". Processing conducted by the invention will not be as effective if different representations are used to represent a given corporate entity, such as "Origin" and "Aurigin, Incorporated" to represent Aurigin.

As noted above, the invention preferably receives patent data (herein called "raw data") from national patent offices. Such patent data often includes different representations of a given corporate entity. Accordingly, corporate entity related processing over such data is not entirely effective.

The invention solves this problem by processing assignee data contained in patent raw data. Such processing, called assignee name processing, is depicted in a flowchart 7402 of FIG. 74.

In step 7406, a normalized assignee name is selected for an entity. For example, the name "Aurigin" may be selected as a normalized assignee name for the corporate entity "Aurigin, Inc.".

In step 7408, the patent raw data is analyzed to identify all name representations for the corporate entity being considered. For example, in reviewing the data set, the following representations for Aurigin, Inc. may be identified:

Aurigin
Origin
Aurigin, Inc.
Aurigin, Incorporated
Origin, Inc.
Origin, Incorporated

Step 7408 may be performed manually, automatically, or a combination thereof. A variety of factors are considered to determine whether or not two representations refer to the same company or two independent and unrelated companies, such as but not limited to company name, company address, etc.

In step 7410, the normalized assignee name selected in step 7406 is linked to the name representations identified in step 7408.

Preferably, step 7410 is performed by replacing instances of the name representations appearing in database tables with the normalized assignee name. Preferably, the patent raw data is not modified.

The processing depicted in FIG. 74 is not limited to assignee name processing, but is instead applicable whenever there is a need that representations

of a given object be consistent throughout a data set. For example, with regard to patents, for example, the processing of FIG. 74 can be applied to inventor names, law firms, patent examiners, classes and subclasses (that change over time), any other bibliographic data, etc.

Input of Arbitrary and Potentially Diverse Data Objects

The present invention enables the automatic or semi-automatic input into the system of data objects of any format and/or content. Such data objects are stored in databases of the system. The invention allows the display and annotation of the data objects. The data objects can also be searched. The invention supports other processing of the data objects, as described herein, and as also described in "System, Method, and Computer Program Product for Managing and Analyzing Intellectual Property (IP) Related Transactions," Ser. No. 09/138,368, filed August 21, 1998; "Using Hyperbolic Trees to Visualize Data Generated By Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/921,369, filed August 29, 1997; and "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, filed June 2, 1997, all of which are incorporated herein by reference in their entireties.

In embodiments of the invention, information is extracted from the data objects. The extracted information is also stored. Preferably, the extracted information is stored in relational databases, data mining databases, or other types of databases. This is similar to storing bibliographic and other information from patents in relational databases, as described above, and as further described in U.S. Patent Application "System, Method and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," Ser. No. 08/867,392, filed June 2, 1997, which is herein incorporated by reference in its entirety. The extracted data that is stored in the relational or other databases can be used for

sophisticated data mining and other processing related to the underlining data objects, as described herein.

Preferably, the format of data objects that are input into the system is at least partially regular and consistent. By this, it is meant that for a given data object type, a given piece of information (such as the document title) is always located at substantially the same position in data objects of the data object type. The invention makes use of the regular and consistent arrangement of data in data objects to extract data from the data objects. The extracted data is stored in relational databases, data mining databases, or other types of databases.

This operation of the invention is represented by a flow chart 8402 in FIG. 84. It is noted that the steps of flowchart 8402 are performed for a particular data object type, which shall herein be referred to as the "current data object type" for reference purposes.

In step 8404, the current data object type is analyzed to identify information of interest 8506 contained in data objects of the current data object type. The identified information of interest 8506 represents information that it is be extracted from data objects of the current data object. For example, if the current data object type is U.S. patents, then the information of interest 8506 identified in step 8404 may include the patent number, the inventors, the assignee, the cited references, the number of claims, the number of figures, etc.

The information of interest 8506 of the current data object type can be identified in step 8404 by a review of an example data object 8502 of the current data object type (see FIG. 85). Alternatively, the information of interest 8506 can be identified in step 8404 by other means. For example, a data object standard 8504 that characterizes/defines/describes the current data object type, and that specifies the data content and the locations of the data content in data objects of the current data object type, can be reviewed to identify the information of interest in step 8404.

In step 8406, the system identifies where the information of interest 8506 (identified in step 8404) is located in data objects of the current data object type. Again, step 8406 can be performed by a review of an example data object 8502 of the current data object, or by review of a data object standard 8504 that characterizes the current data object type.

In step 8408, one or more new database tables are created, if necessary, to store the data that it to be extracted from data objects of the current data object type. It is noted that step 8408 is optional, in that database tables for storing the extracted information may already exist.

In step 8410, one or more database tables 8510 (one or more of which may have been created in 8408) are selected. The selected database tables 8510 will store the information extracted from data objects of the current data object type. The database tables 8510 selected in step 8410 are represented, for example, as relational database tables 8510 in FIG. 85.

In step 8412, the columns (fields) in the selected database tables 8510 where the information extracted from the data objects of the current data object type is to be stored are selected. This is represented in FIG. 85, for example, where it is specified that data in location 8508A is stored in column 8512 of relational database 8510. Similarly, data in locations 8508B, 8508C and 8508D of the data objects of the current data object type are stored in columns 8514, 8516 and 8518, respectively, of the relational database tables 8510.

After completion of step 8412, the system is prepared to receive and extract data from data objects of the current data object type. Accordingly, in step 8414, a data object of the current data object type is received.

In step 8416, the system determines the type of the received data object. In particular, the system in step 8416 determines that the received data object is of the current data object type. The system can determine the type by reviewing information that accompanied the data object when it was received in step 8414. In particular, in step 8414, an indication of the type of the data object may also

have been provided to the system. Alternatively, the system can analyze the received data object to determine its type.

In step 8418, the system extracts information from the received data object in accordance with the data object's type. Preferably, this extraction operation does not change or otherwise affect the received data object. The system knows where to extract information of interest 8506 by reference to the locations 8508 identified in step 8406.

In step 8420, the system stores the extracted information in columns of the database tables 8510 associated with the current database type, as determined in steps 8410 and 8412.

In step 8422, the system stores the data object itself.

Tool Box Embodiments

A variety of widgets (components) of the invention are described herein. The invention is directed to tool box embodiments where any number of these widgets are packaged as separately accessible components that can be added to other applications, or used to customize the system described herein. For example, and without limitation, an extension of the user interface includes the ability to "mix and match" specific widgets or components associated with the user interface. Consider FIGS. 16 and 75, and the "kind pane" (7504). These could be packaged as separately accessible components of the API and made available as a "toolkit". The same is true with all other components of the invention. Users could build their own user interface or other applications -- perhaps by embedding the context browser into other applications, such as a spreadsheet in Microsoft Excel.

Example Implementation

In an embodiment of the present invention, the system and components of the present invention described herein are implemented using well known computers, such as a computer 8602 shown in FIG. 86. The computer 8602 can be any commercially available and well known computer capable of performing the functions described herein, such as computers available from International Business Machines, Apple, Silicon Graphics Inc., Sun, HP, Dell, Compaq, Digital, Cray, etc.

The computer 8602 includes one or more processors (also called central processing units, or CPUs), such as a processor 8606. The processor 8606 is connected to a communication bus 8604. The computer 8602 also includes a main or primary memory 8608, preferably random access memory (RAM). The primary memory 8608 has stored therein control logic (computer software), and data.

The computer 8602 also includes one or more secondary storage devices 8610. The secondary storage devices 8610 include, for example, a hard disk drive 8612 and/or a removable storage device or drive 8614. The removable storage drive 8614 represents a floppy disk drive, a magnetic tape drive, a compact disk drive, an optical storage device, tape backup, ZIP drive, JAZZ drive, etc.

The removable storage drive 8614 interacts with a removable storage unit 8616. As will be appreciated, the removable storage unit 8616 includes a computer usable or readable storage medium having stored therein computer software (control logic) and/or data. The removable storage drive 8614 reads from and/or writes to the removable storage unit 8616 in a well known manner.

Removable storage unit 8616, also called a program storage device or a computer program product, represents a floppy disk, magnetic tape, compact disk, optical storage disk, ZIP disk, JAZZ disk/tape, or any other computer data storage device. Program storage devices or computer program products also include any device in which computer programs can be stored, such as hard drives, ROM or memory cards, etc.

In an embodiment, the present invention is directed to computer program products or program storage devices having software that enables the computer 8602 to perform any combination of the functions described herein.

Computer programs (also called computer control logic) are stored in main memory 8608 and/or the secondary storage devices 8610. Such computer programs, when executed, enable the computer 8602 to perform the functions of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 8606 to perform the functions of the present invention. Accordingly, such computer programs represent controllers of the computer 8602.

The computer 8602 also input/output/display devices 8622, such as monitors, keyboards, pointing devices, etc.

The computer 8602 further includes a communication or network interface 8618. The network interface 8618 enables the computer 8602 to communicate with remote devices. For example, the network interface 8618 allows the computer 8602 to communicate over communication networks, such as LANs, WANs, the Internet, etc. The network interface 8618 may interface with remote sites or networks via wired or wireless connections. The computer 8602 receives data and/or computer programs 8620 via the network interface 8618. The electrical/magnetic signals having contained therein data and/or computer programs 8620 received or transmitted by the computer 8602 via interface 8618 also represent computer program product(s).

The invention can work with software, hardware, and operating system implementations other than those described herein. Any software, hardware, and operating system implementations suitable for performing the functions described herein can be used.

Conclusion

While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What Is Claimed Is:

1. A computer implemented method of enabling review of patents in an electronic patent shoe, comprising the steps of:

(1) causing generation of an electronic patent shoe comprising a plurality of patents;

(2) causing display of a list of patents in said electronic patent shoe;

(3) causing display of image data representative of at least a portion of a patent from said electronic patent shoe;

(4) causing, pursuant to a command to view a next patent, retrieval and display of at least image data representative of at least a portion of said next patent; and

(5) causing, pursuant to a command to view a previous patent, retrieval and display of at least image data representative of at least a portion of said previous patent.

2. The method of claim 1, further comprising the steps of:

(6) determining whether requested information pertaining to a patent is locally available; and

(7) if said requested information is not locally available, then providing an option to preview said requested information; and

(8) if said option is selected, then retrieving and displaying at least a portion of said requested information.

3. A method of context data processing, comprising the steps of:

(1) selecting one or more contexts each including one or more attributes and a plurality of data objects that satisfy said attributes;

(2) displaying a list of data objects contained in said contexts; and

(3) processing a plurality of said data objects contained in said contexts.

4. The method of claim 3, wherein said contexts are groups.

5. The method of claim 3, wherein each of said contexts is associated with a data object type and comprises data objects of said data object type.

6. The method of claim 3, wherein said data objects are of interest to a corporate entity.

7. The method of claim 6, wherein said data objects are patents, and wherein step (3) comprises the step of:

(a) performing patent-centric processing of said data objects.

8. The method of claim 7, wherein step (a) comprises the step of:

generating a patent citation tree comprising patents and non-patent data objects that (i) are cited by a base patent, or (ii) cite a base patent.

9. The method of claim 8, further comprising the step of:

visualizing said patent citation tree as a hyperbolic tree.

10. The method of claim 9, wherein each node of said hyperbolic tree comprises state information including display attribute information.

11. The method of claim 7, wherein step (a) comprises the step of:

generating a patent claim tree comprising nodes corresponding to claims in one of said patents, and links between said nodes indicative of dependencies among said claims.

12. The method of claim 11, further comprising the step of:

displaying at least a portion of a claim corresponding to one of said nodes when a selection device is positioned proximate to said one of said nodes.

13. The method of claim 11, further comprising the step of:

displaying a dependent claim, a base claim of said dependent claim, and any intervening claims between said dependent claim and said base claim upon appropriate user command.

14. The method of claim 3, further comprising the steps of:

(4) generating a data object family tree comprising a plurality of data objects and indicative of relationships between said data objects; and

(5) displaying said data object family tree.

15. The method of claim 14, wherein step (5) comprises the step of:

displaying said data object family tree as a hyperbolic tree.

16. The method of claim 3, further comprising the steps of:

(4) generating at least one annotation.

17. The method of claim 16, wherein step (4) comprises the step of:

generating a document annotation that is linked to at least a portion of at least one of said data objects, and that has a scope of a group containing said at least one of said data objects.

18. The method of claim 17, wherein said document annotation is linked to at least a portion of image data related to said at least one of said data objects.

19. The method of claim 16, wherein step (4) comprises the step of:

generating a group annotation that has a scope of a group active when said group annotation was created.

20. The method of claim 16, wherein step (4) comprises the step of:

generating a data object type annotation that has a scope of a data object type active when said data object type annotation was created.

21. The method of claim 16, wherein step (4) comprises the step of:

generating a case annotation that is linked to at least a portion of at least one of said data objects, and that has a scope of a case active when said case annotation was created.

22. The method of claim 16, wherein step (4) comprises the step of:

generating an enterprise annotation that is linked to at least a portion of at least one of said data objects, and that is always accessible to users with appropriate authorization.

23. A system for processing data, comprising:

an intellectual property asset manager (IPAM);

a plug-in manager coupled to said IPAM; and

at least one plug-in coupled to said plug-in manager.

24. The system of claim 23, further comprising:

at least one external data processing component coupled to said plug-in.

25. The system of claim 24, wherein said external data processing component displays data using at least graphs.

26. The system of claim 24, wherein said external data processing component displays data using at least maps.

27. The system of claim 24, wherein said plug-in manager has a first application programming interface (API), and said external data processing component has a second API.

28. The system of claim 27, wherein said plug-in comprises:

means for translating messages from said plug-in manager to said external data processing component to a format conforming to said second API; and

means for translating messages from said external data processing component to said plug-in manager to a format conforming to said first API.

29. A method of assignee name processing, comprising the steps of:

(1) selecting a normalized assignee name for an entity;

(2) identifying name representations of said entity in a data set; and

(3) linking said name representations to said normalized assignee name.

30. The method of claim 29, wherein step (3) comprises the steps of:

substituting at least some instances of said name representations with said normalized assignee name.

31. A method of context data processing, comprising the steps of:

(1) accessing databases of patents and patent bibliographic information;

(2) accessing one or more contexts, each of said one or more contexts comprising any number of patents represented in said databases; and

(3) automatically performing patent centric and context oriented processing of at least some of said patents in at least one of said one or more contexts.

32. The method of claim 31, further comprising the step of:

(4) annotating a patent represented in said databases.

33. The method of claim 32, wherein step (4) comprises the step of:

annotating said patent using an annotation form.

34. The method of claim 31, wherein step (3) comprises:

generating and displaying data object families.

35. A system for processing data, comprising:

an intellectual property asset manager (IPAM); and

at least one external data processing component coupled to said IPAM to expand functionality of said IPAM, wherein said external data processing component displays data from said IPAM using at least one of graphs and maps.

36. The system of claim 35, further comprising:

means for accessing databases of patents and patent bibliographic information;

means for accessing one or more groups, each of said one or more groups comprising any number of said patents represented in said databases; and

means for automatically performing patent centric and group oriented processing of at least some of said patents in at least one of said one or more groups.

37. The system of claim 35, further comprising:

means for generating and displaying data object families.

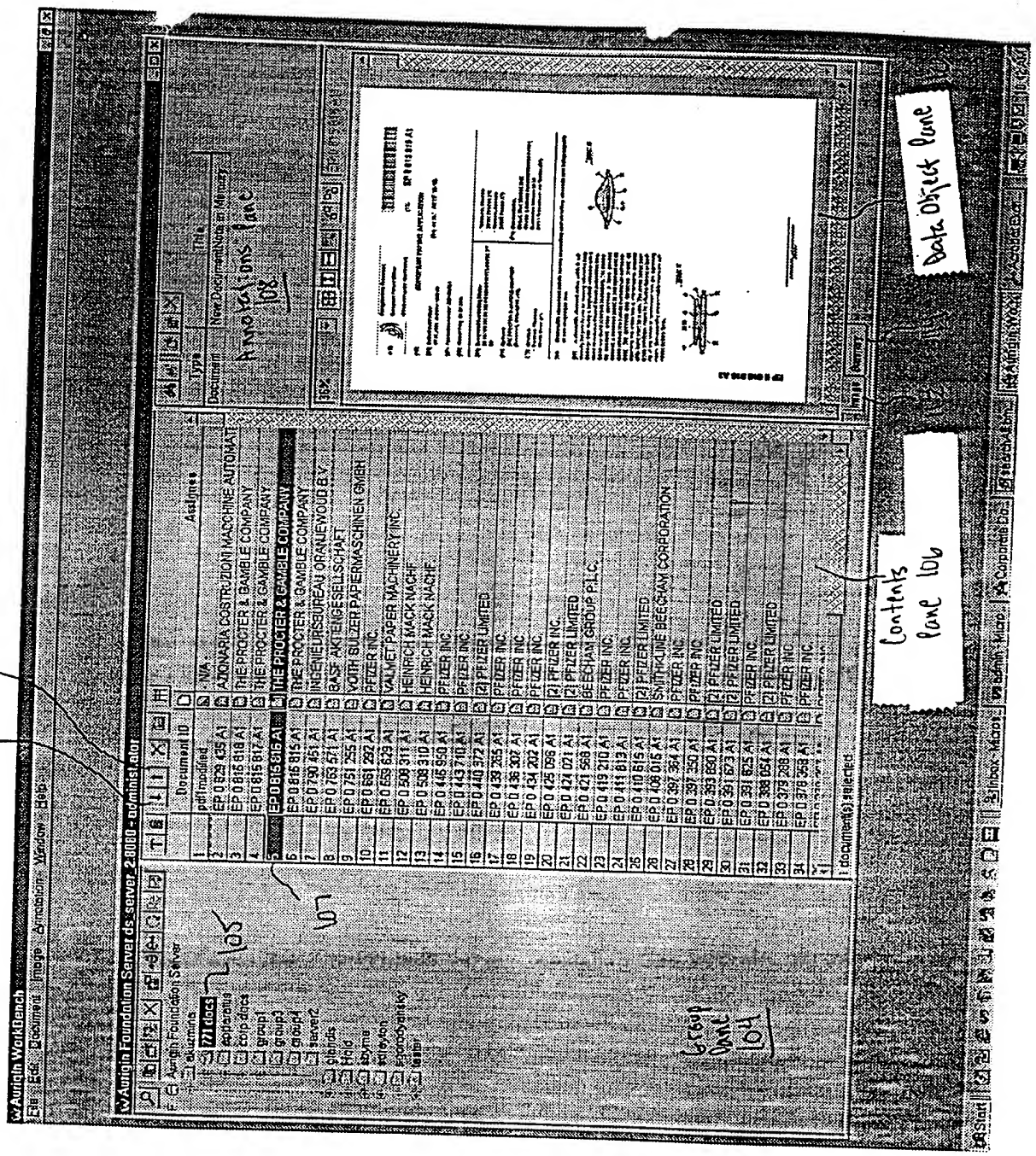

Client Browser (Console)
102
Base Object Pane
Contents Pane 106
Annotations 108
Type
Document
Name
Author
Title
Document ID
pdf modified
Hold
Start

FIG. 1

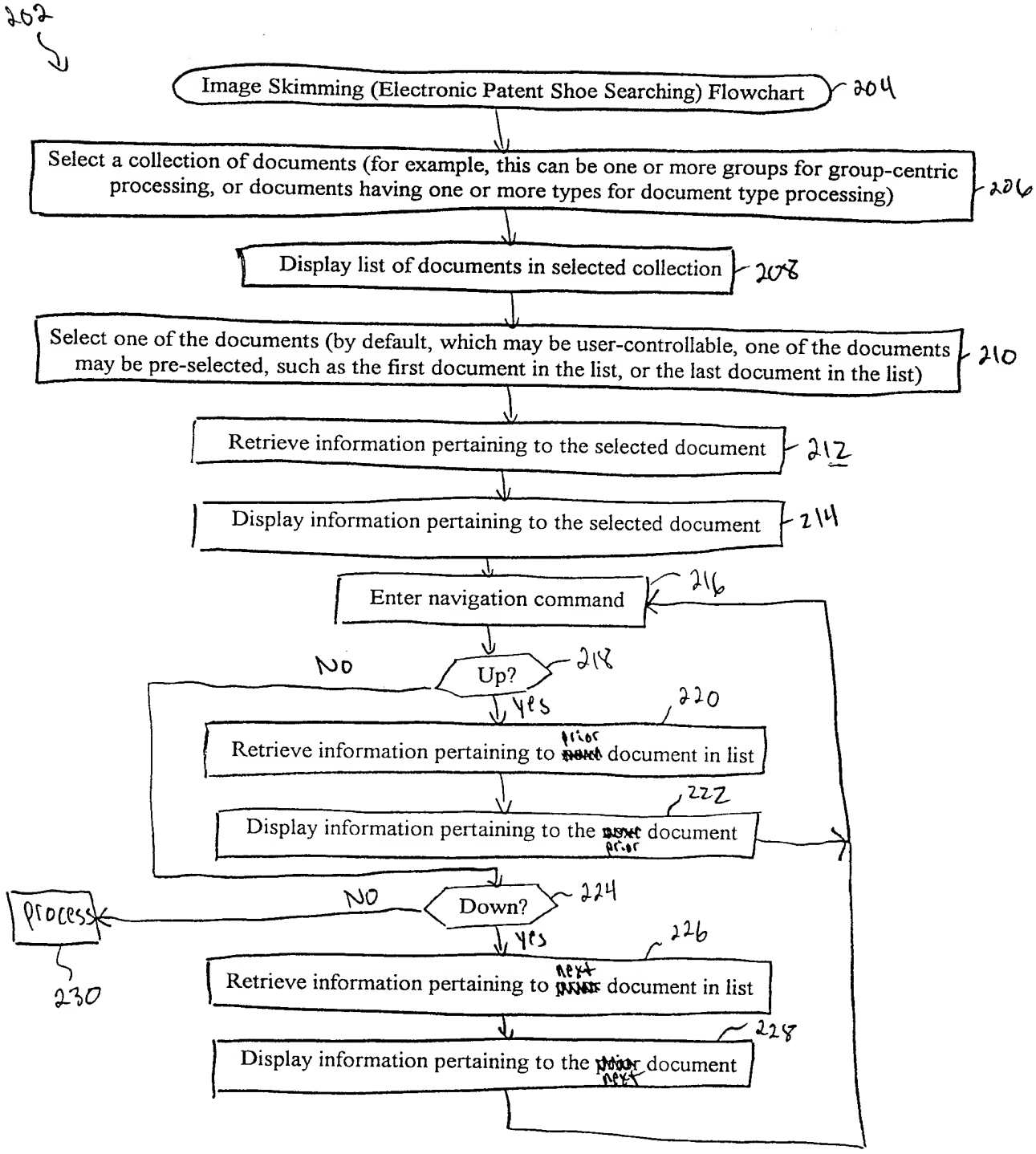

202
Image Skimming (Electronic Patent Shoe Searching) Flowchart
204
Select a collection of documents (for example, this can be one or more groups for group-centric processing, or documents having one or more types for document type processing)
206
Display list of documents in selected collection
208
Select one of the documents (by default, which may be user-controllable, one of the documents may be pre-selected, such as the first document in the list, or the last document in the list)
210
Retrieve information pertaining to the selected document
212
Display information pertaining to the selected document
214
Enter navigation command
216
Up?
218
NO
Yes
Retrieve information pertaining to prior document in list
220
Display information pertaining to the prior document
222
Down?
224
NO
Yes
Process
230
Retrieve information pertaining to next document in list
226
Display information pertaining to the next document
228

FIG. 2

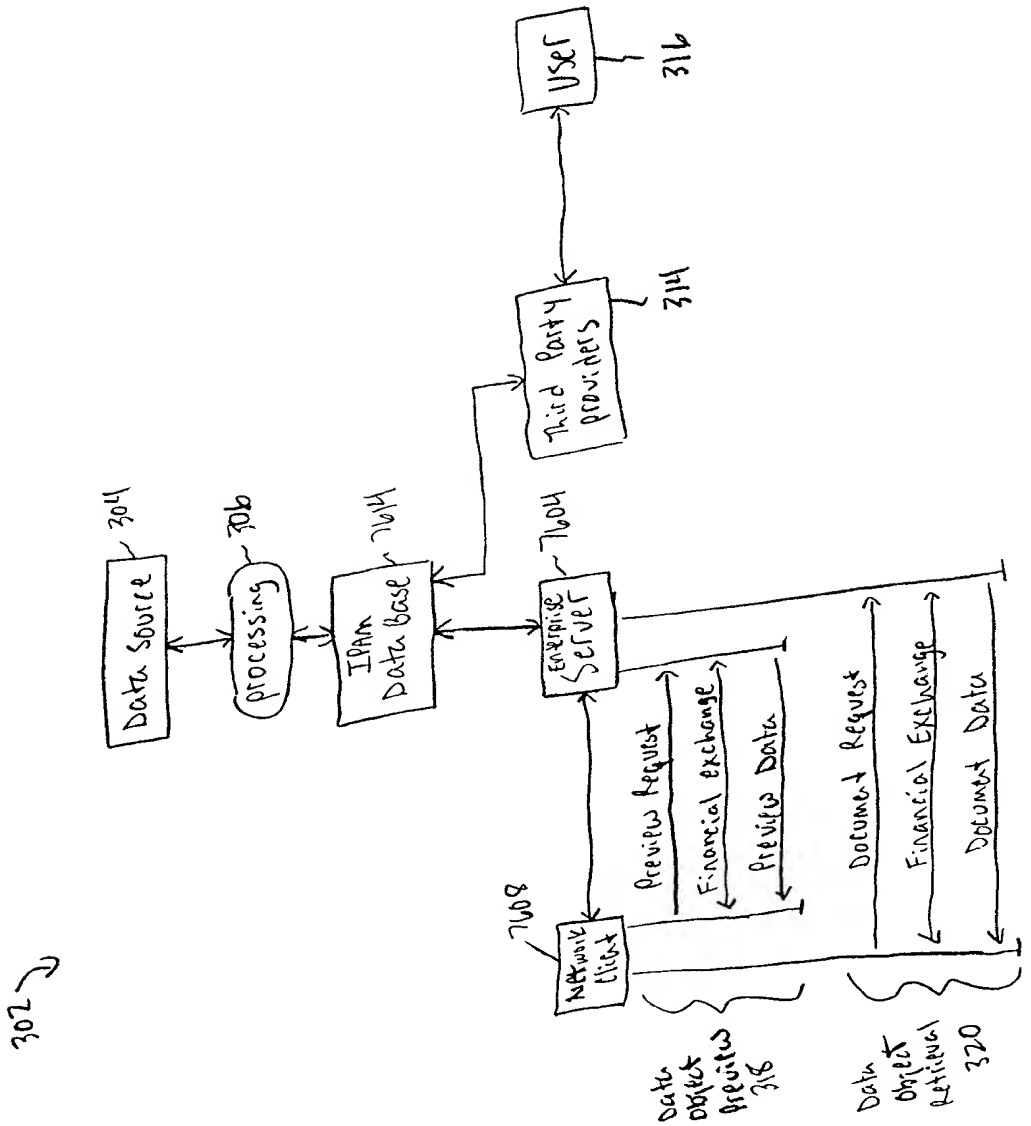

User
316
Third Party providers
314
Data Source
304
processing
306
IDAM Data Base
308
Enterprise Servers
310
Network Client
308
Previous Request
Financial Exchange
Previous Data
Document Request
Financial Exchange
Document Data
Data Object Previous
318
Data Object Retrieval
320
302

FIG. 3

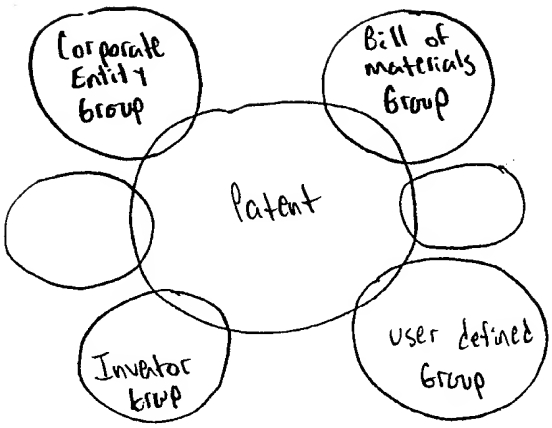

Corporate Entity Group
Bill of materials Group
Patent
Inventor Group
user defined Group

FIG. 4

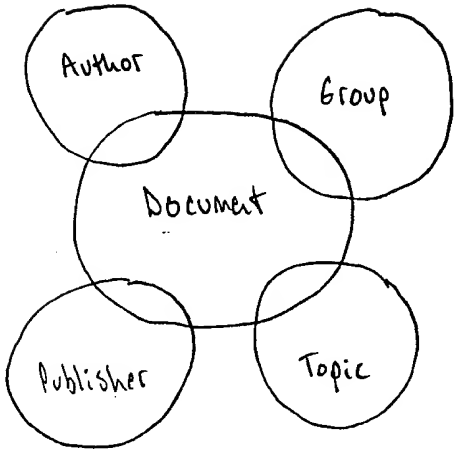


FIG. 5

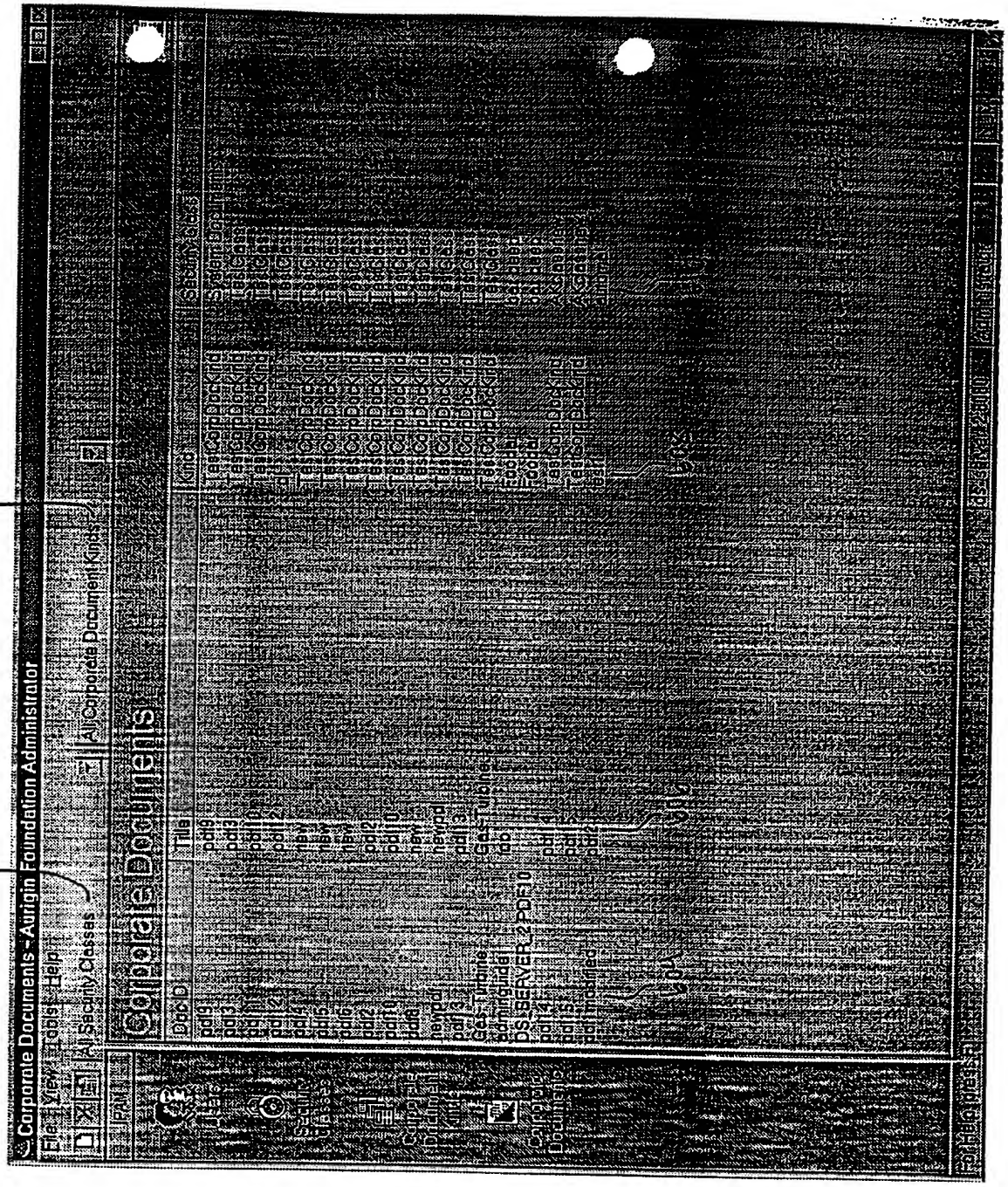
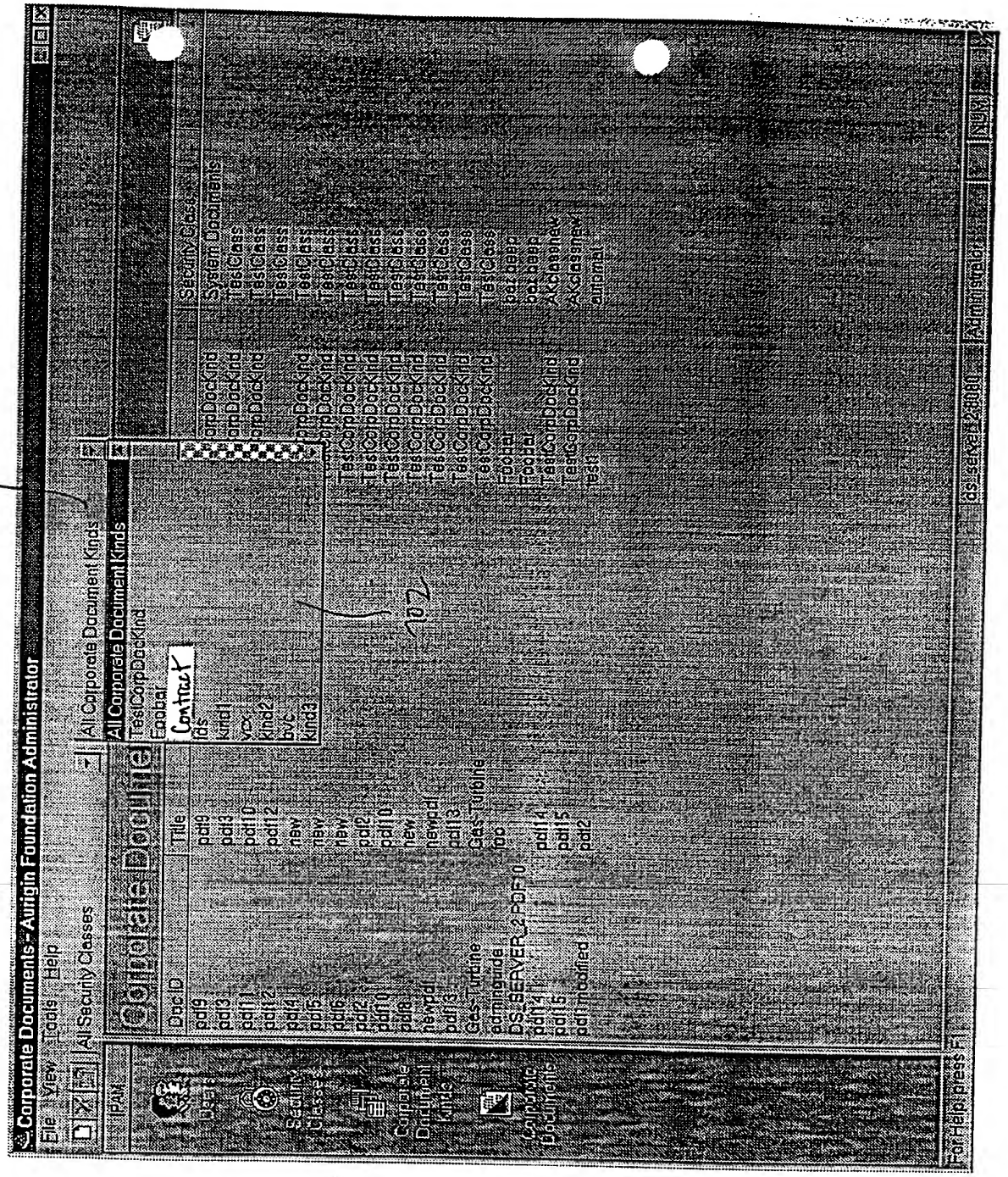

Corporate Documents - Autogin Foundation Administrator
File View Tools Help
All Security Classes
All Corporate Document Kinds
Corporate Documents
Doc ID
Title
Kind
Security Class
Gas Turbine
Document
Corporate Document
Foundation Ready
206
204
202

Fig. 9


Corporate Documents - Aurigin Foundation Administrator
File View Tools Help
All Security Classes
All Corporate Document Kinds
All Corporate Document Kinds
TestCorpDocKind
Contract
DocID
Title
Security Class
System Documents
TestClass
For Help, press F1
Corporate Document Window 102
116
102
FIG. 7

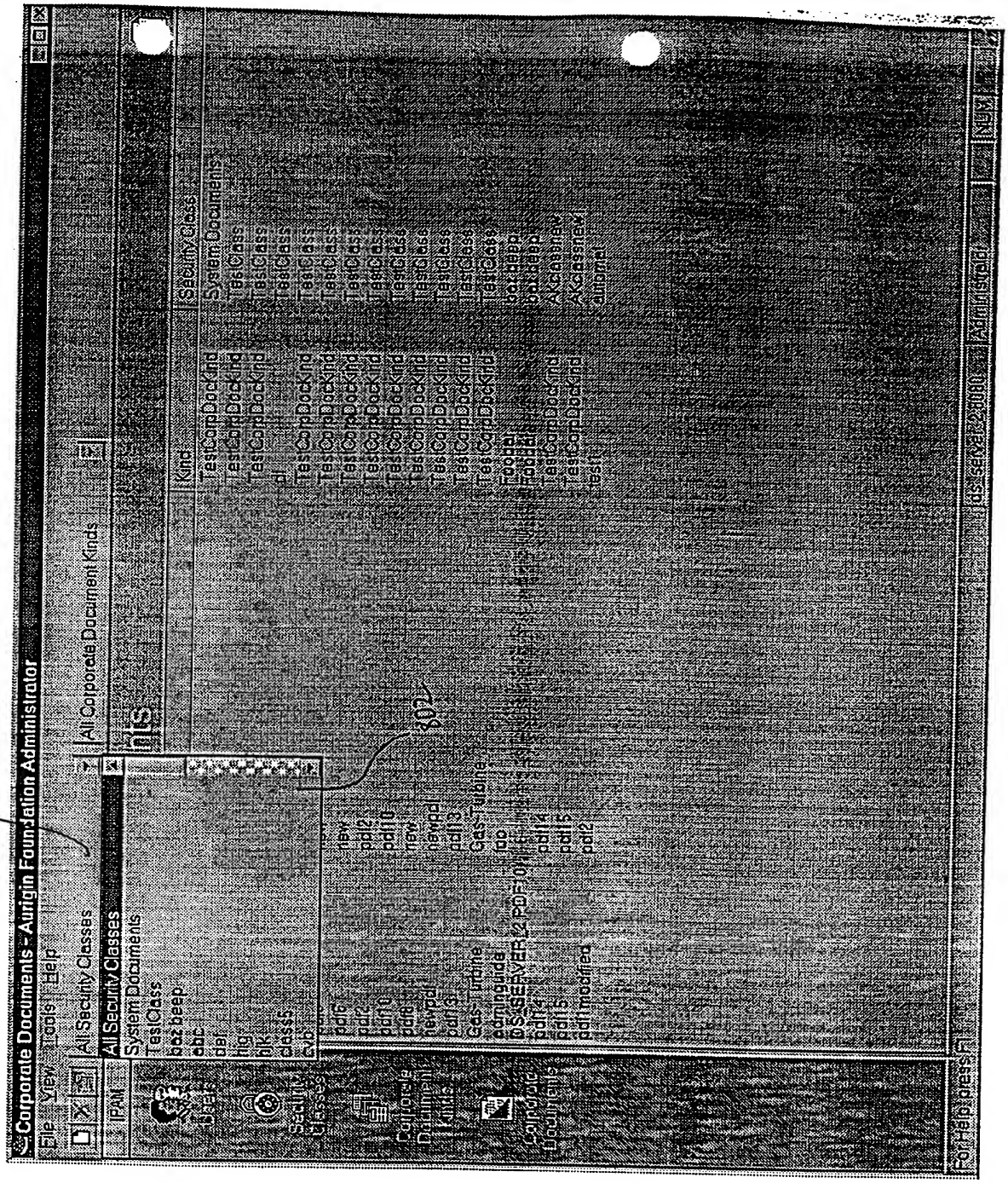


Fig. 8

Corporate Document Instances 702

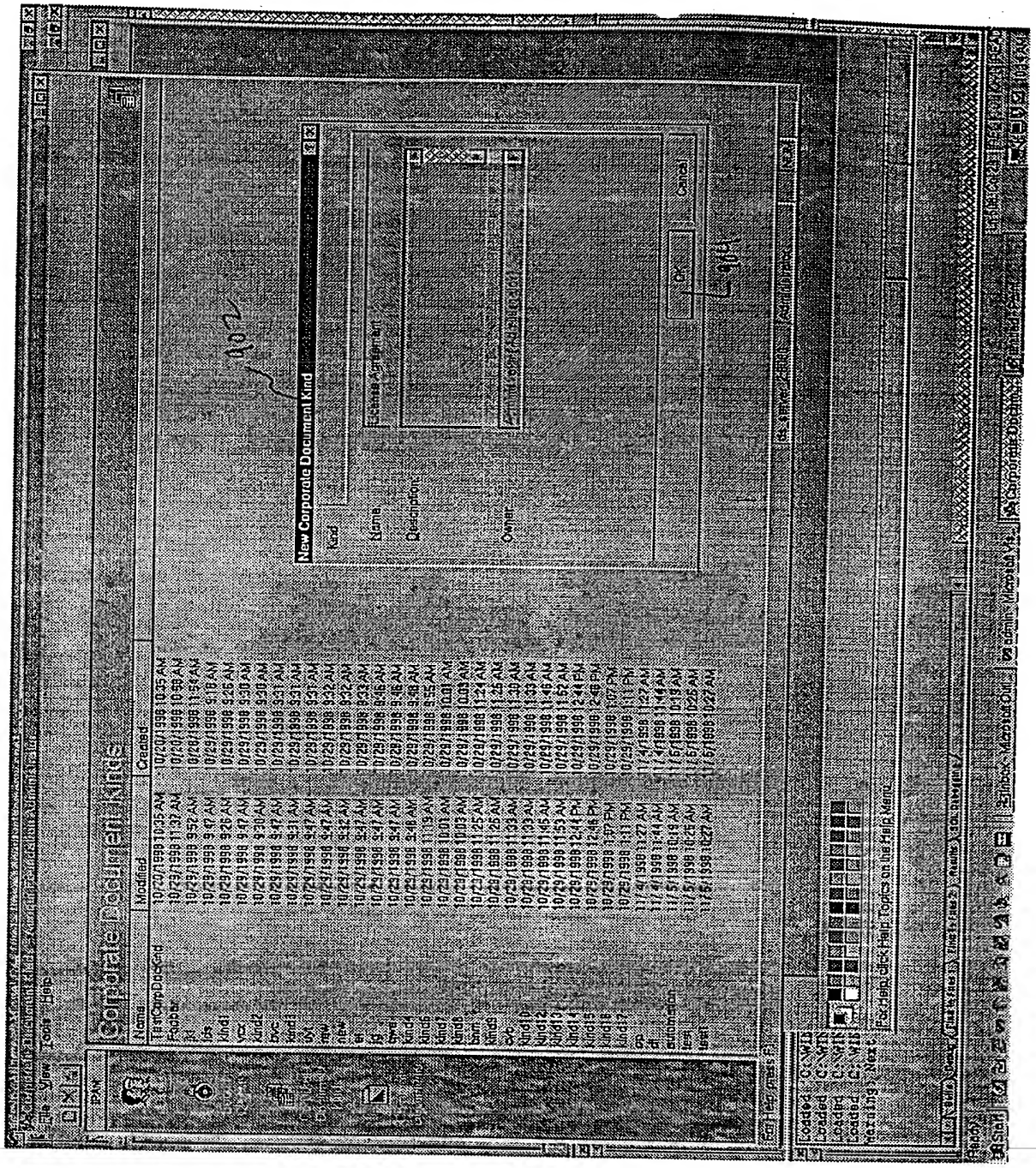

New Corporate Document Kind
Name
Description
Owner
OK
Cancel

FIG. 9

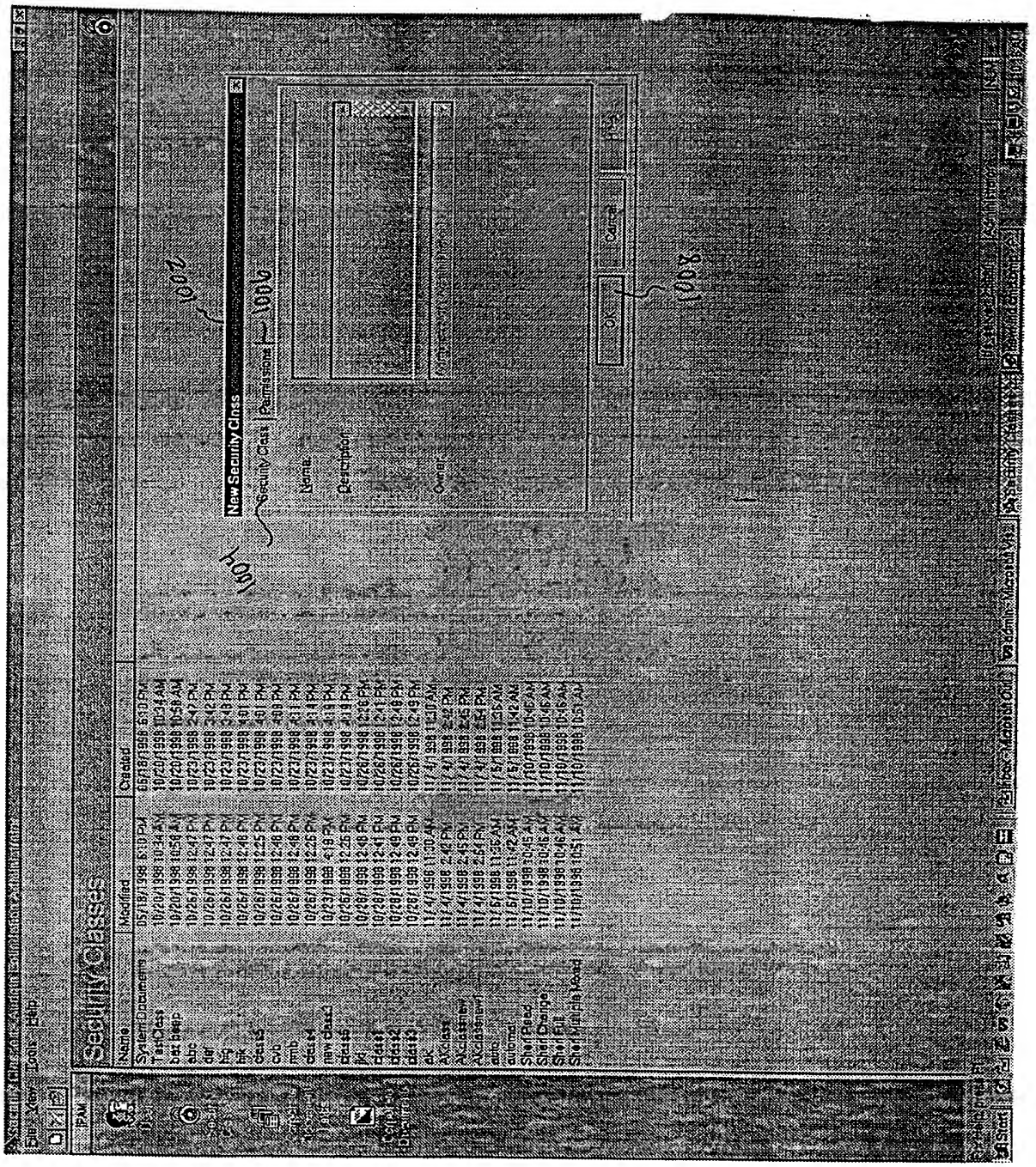

New Security Class
Security Class
Permissions
Name
Description
Owner
Cancel
OK
100a
100b
100c

FIG. 10

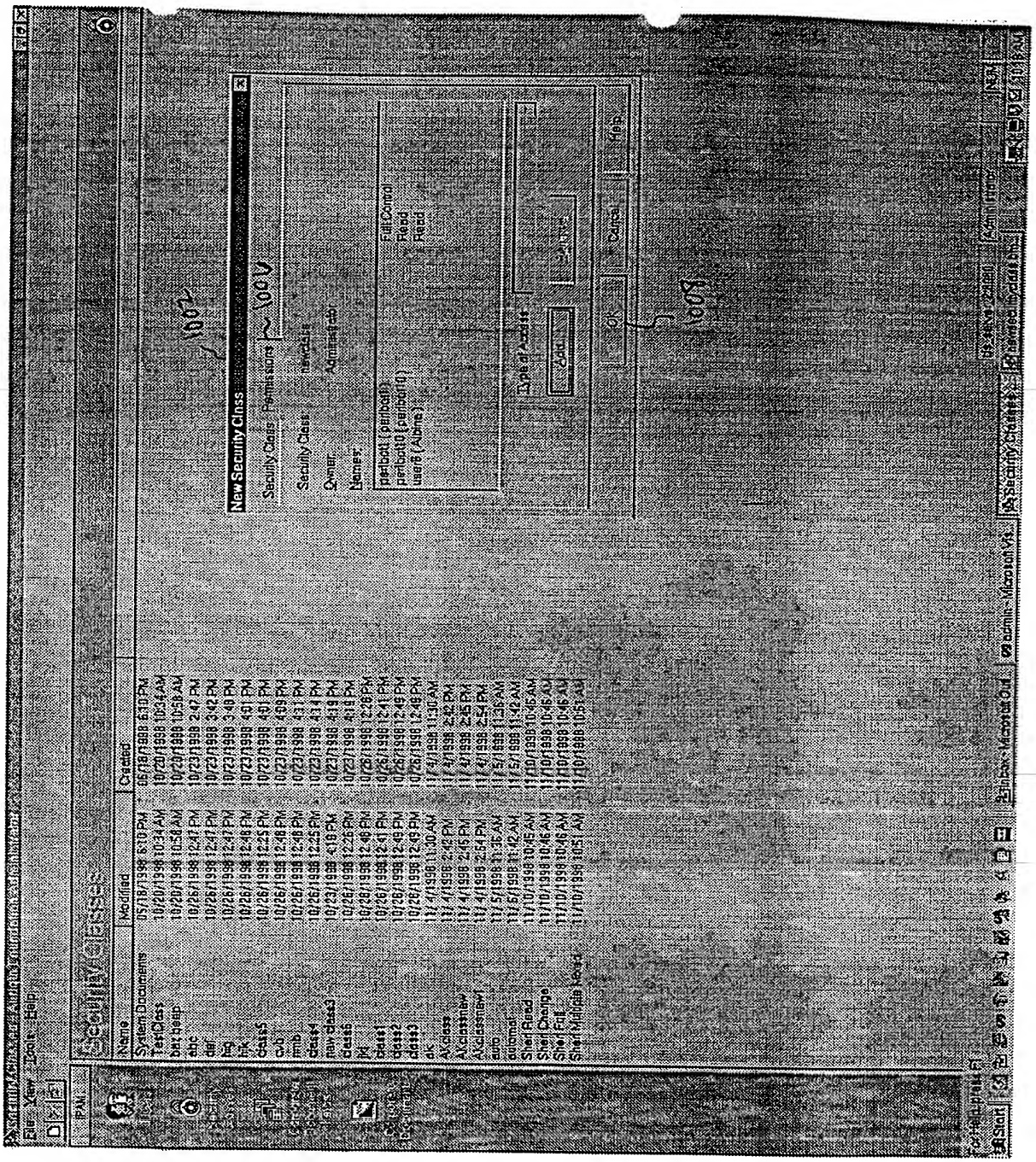

New Security Class
Security Class Permissions
Security Class newclass
Owner Administrator
Owner
permit (permit)
permit0 (permit0)
user9 (Alpha)
FullControl
Read
Read
Add...
OK
Cancel
Help
Type of Access
Add...
OK
Cancel
Help
1002
1004

FIG 11

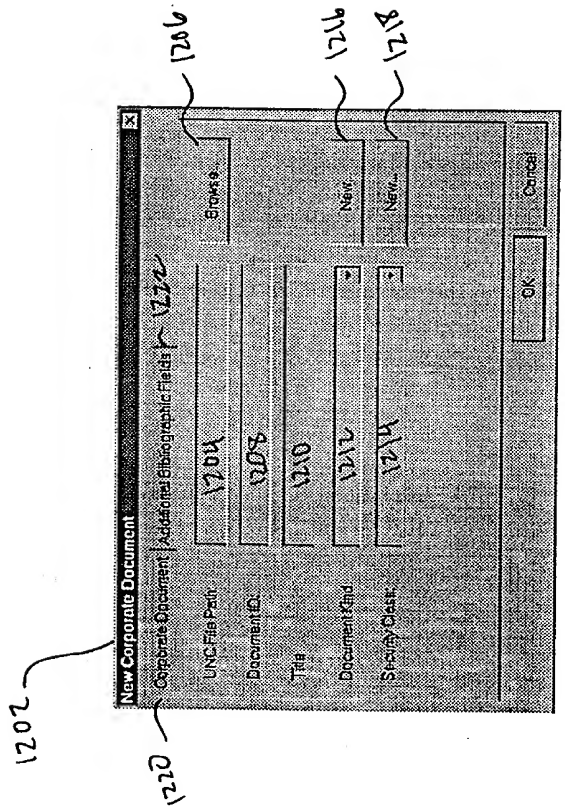
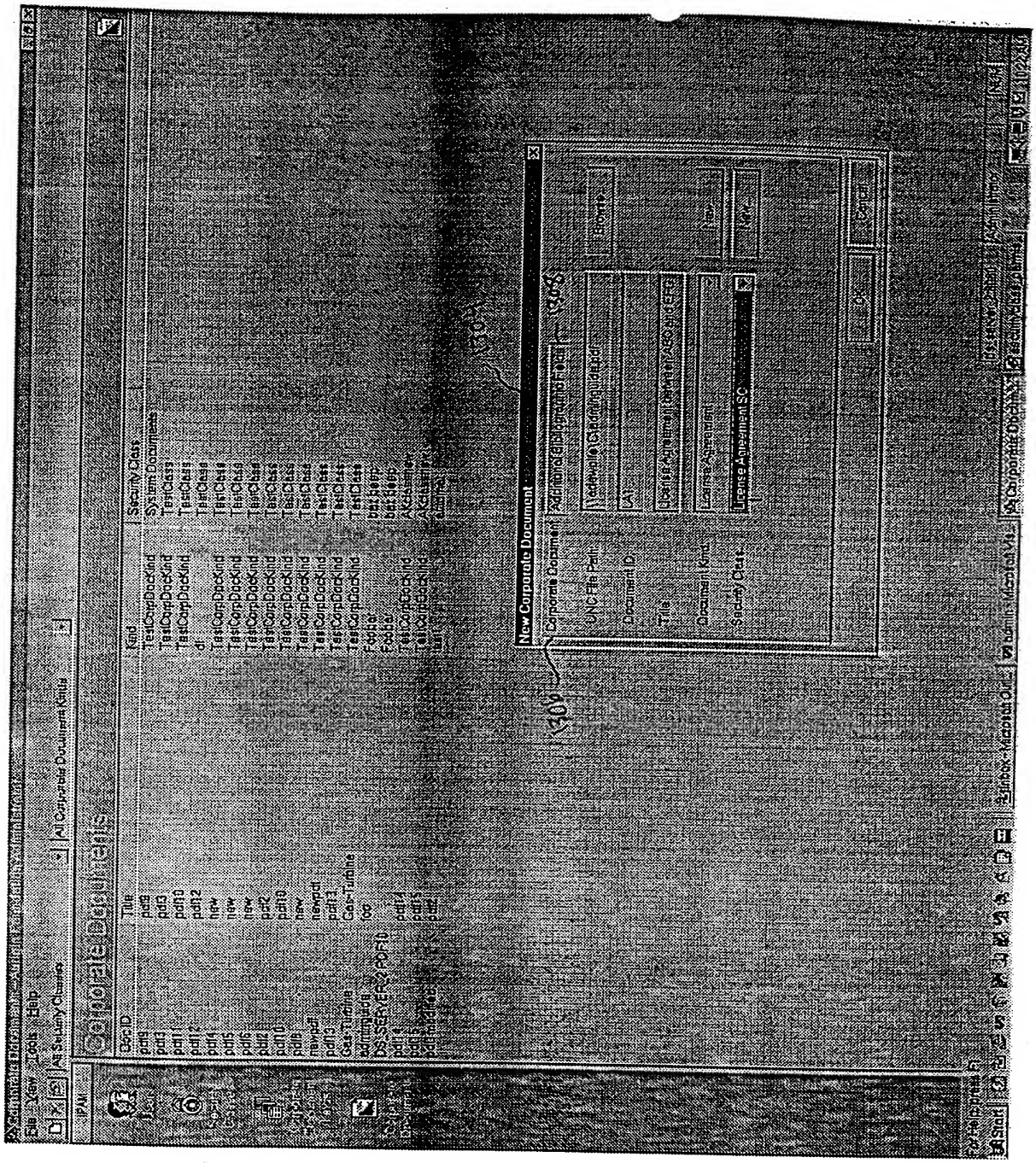

New Corporate Document
Corporate Document
Additional Bibliographic Fields
UNC File Path
Document ID
Title
Document Kind
Security Class
Browse...
New...
New...
OK
Cancel
1202
1220
1222
1206
1216
1218

Fig. 12



for parent window 1302

FIG. 13

Corporate Document Window 1302
1304
1304
New Corporate Document
Corporate Document Additional Bibliographic Entries
Author
Source
Publication Date
OK
Cancel

FIG. 14

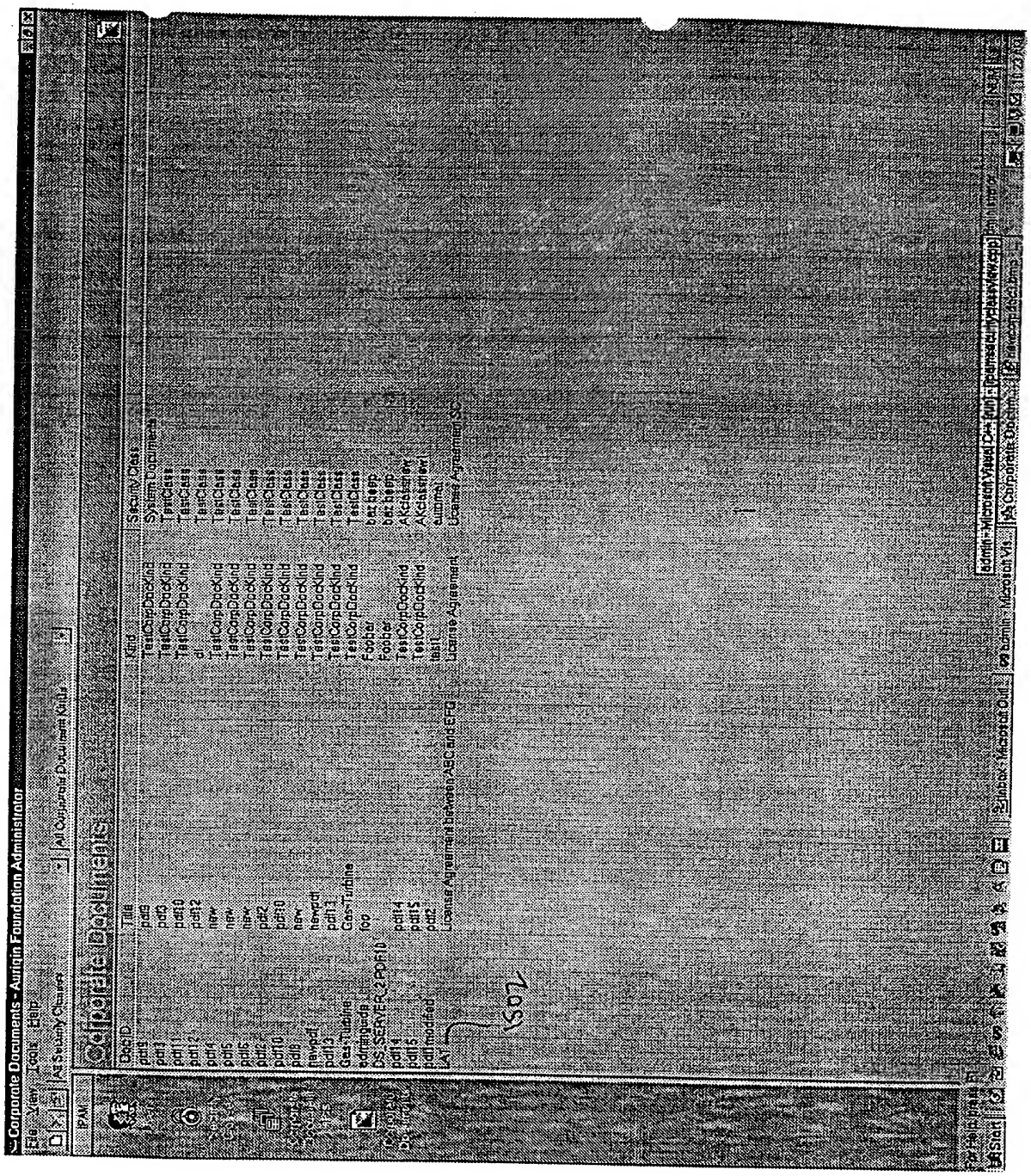


FIG. 15

Corporate Document Window 1302

Annotations Pane 108
Content Browser 102
Document List Pane
Type Entries
Summary Info
Text
Image
Audio
Video
Contents Pane 106
Kind Pane 104
Kind Alignment
Document ID
Type
Start

FIG. 1A

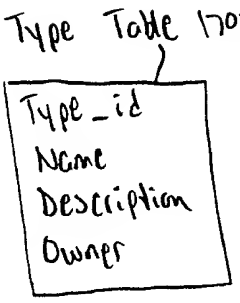

Type Table 1702
Type_id
Name
Description
Owner

FIG. 17

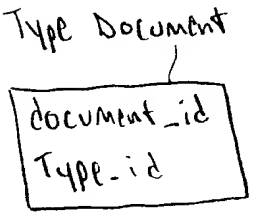

Type Document Xref 1802
document_id
Type_id

FIG. 18

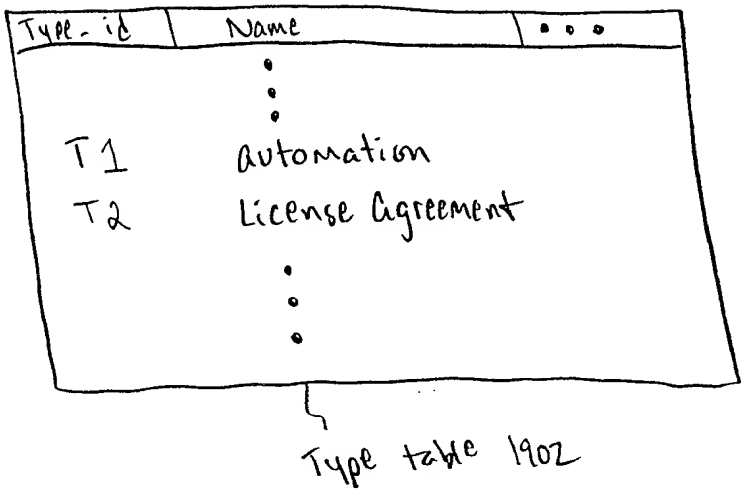

Type_id
Name
T1
Automation
T2
License Agreement
Type table 1902

FIG. 19

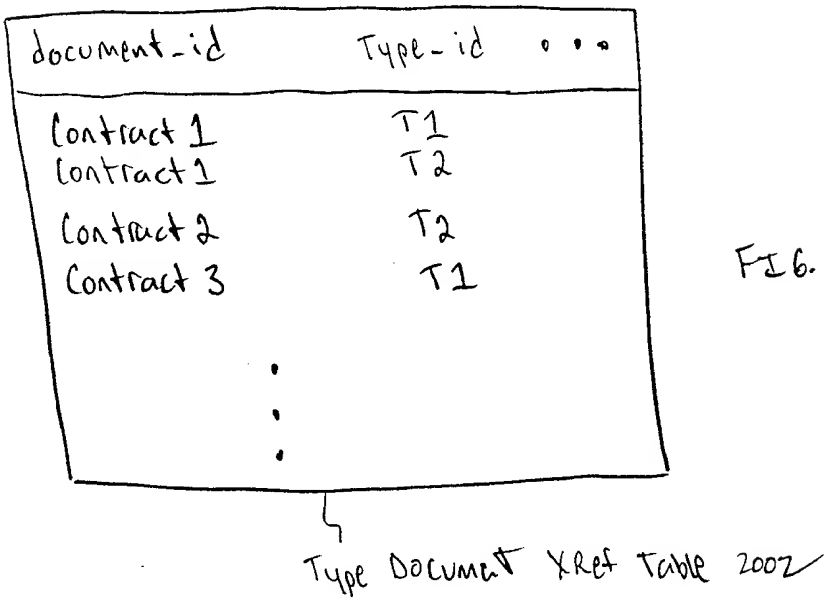

document_id
Type_id
Contract 1
T1
Contract 1
T2
Contract 2
T2
Contract 3
T1
Type Document XRef Table 2002

FIG. 20

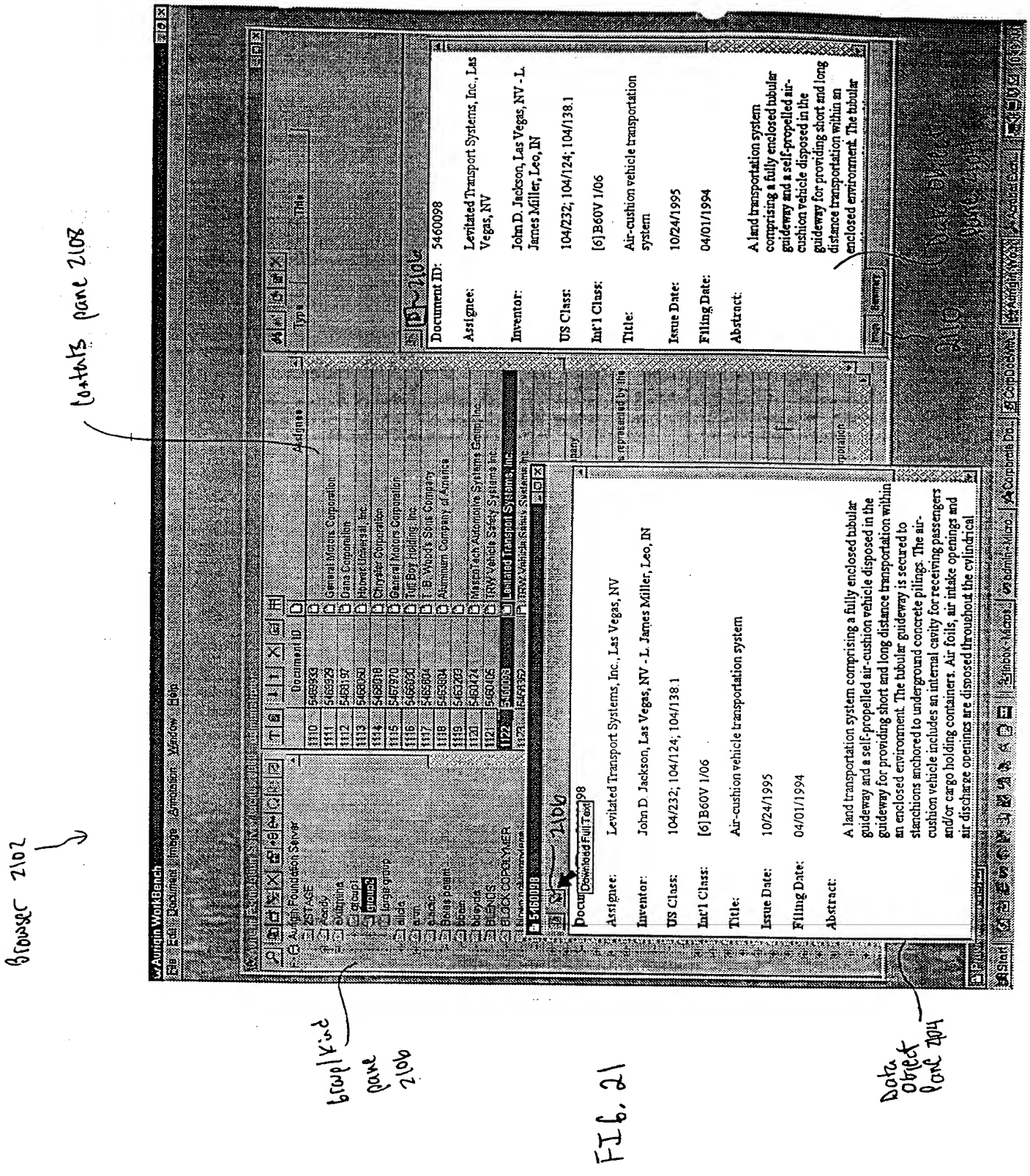

Browser 2102
Contents pane 2108
Group/Find pane 2106
Data Object Pane 2104
Document ID: 5460098
Assignee: Levitated Transport Systems, Inc., Las Vegas, NV
Inventor: John D. Jackson, Las Vegas, NV - L. James Miller, Leo, IN
US Class: 104/232; 104/124; 104/138.1
Int'l Class: [6] B60V 1/06
Title: Air-cushion vehicle transportation system
Issue Date: 10/24/1995
Filing Date: 04/01/1994
Abstract: A land transportation system comprising a fully enclosed tubular guideway and a self-propelled air-cushion vehicle disposed in the guideway for providing short and long distance transportation within an enclosed environment. The tubular

FIG. 21

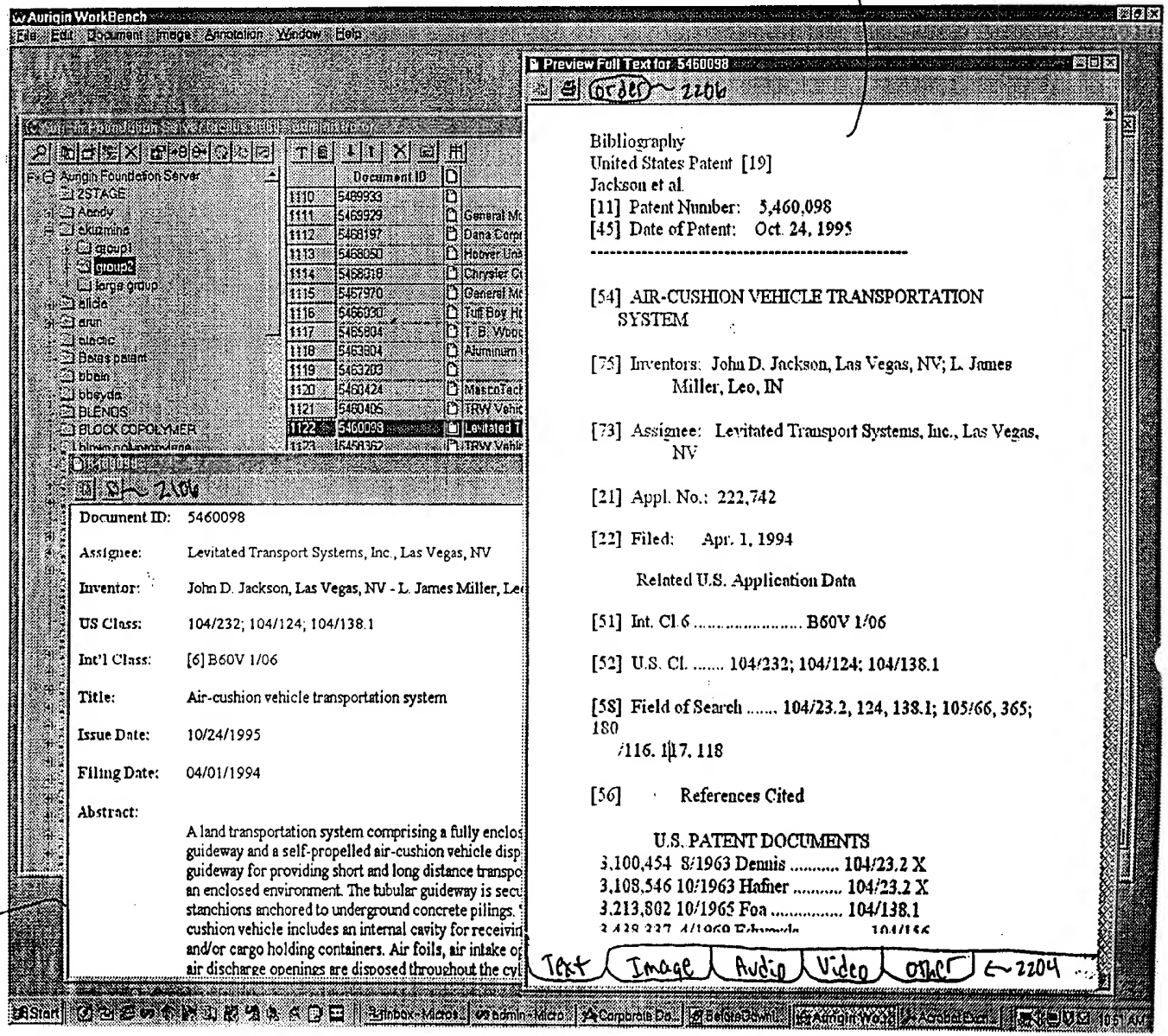

Browser 2102
Preview Window 2202
Aurigin WorkBench
File Edit Document Image Annotation Window Help
Aurigin Foundation Server
2STAGE
Aandy
skimming
group1
group2
large group
allida
erun
Beta's patent
bbain
bbeyda
BLENDS
BLOCK COPOLYMER
Document ID
1110 5489933
1111 5469929
1112 5468197
1113 5468050
1114 5468019
1115 5467970
1116 5466030
1117 5465804
1118 5463804
1119 5463203
1120 5460424
1121 5460405
1122 5460098
Levitated T
Document ID: 5460098
Assignee: Levitated Transport Systems, Inc., Las Vegas, NV
Inventor: John D. Jackson, Las Vegas, NV - L. James Miller, Le
US Class: 104/232; 104/124; 104/138.1
Int'l Class: [6] B60V 1/06
Title: Air-cushion vehicle transportation system
Issue Date: 10/24/1995
Filing Date: 04/01/1994
Abstract:
A land transportation system comprising a fully enclos
guideway and a self-propelled air-cushion vehicle disp
guideway for providing short and long distance transpo
an enclosed environment. The tubular guideway is secu
stanchions anchored to underground concrete pilings.
cushion vehicle includes an internal cavity for receivi
and/or cargo holding containers. Air foils, air intake o
air discharge openings are disposed throughout the cy
Data Object Pane 2104
Preview Full Text for 5460098
Order 2206
Bibliography
United States Patent [19]
Jackson et al.
[11] Patent Number: 5,460,098
[45] Date of Patent: Oct. 24, 1995
[54] AIR-CUSHION VEHICLE TRANSPORTATION SYSTEM
[75] Inventors: John D. Jackson, Las Vegas, NV; L. James Miller, Leo, IN
[73] Assignee: Levitated Transport Systems, Inc., Las Vegas, NV
[21] Appl. No.: 222,742
[22] Filed: Apr. 1, 1994
Related U.S. Application Data
[51] Int. Cl.6 ..... B60V 1/06
[52] U.S. Cl. 104/232; 104/124; 104/138.1
[58] Field of Search 104/23.2, 124, 138.1; 105/66, 365; 180
/116. 117. 118
[56] References Cited
U.S. PATENT DOCUMENTS
3,100,454 8/1963 Dennis ..... 104/23.2 X
3,108,546 10/1963 Hafner ..... 104/23.2 X
3,213,802 10/1965 Foa 104/138.1
Text Image Audio Video Other 2204

FIG. 22

2302

Annotation Type	Attached to a Data Object?	Scope of Annotation			
		Data Object	Case	Group	Type
Document	Yes			X	
Case	Yes		X		
Group	No			X	
Type	No				X
Enterprise	Yes	X			

FIG. 23

Browser 2402

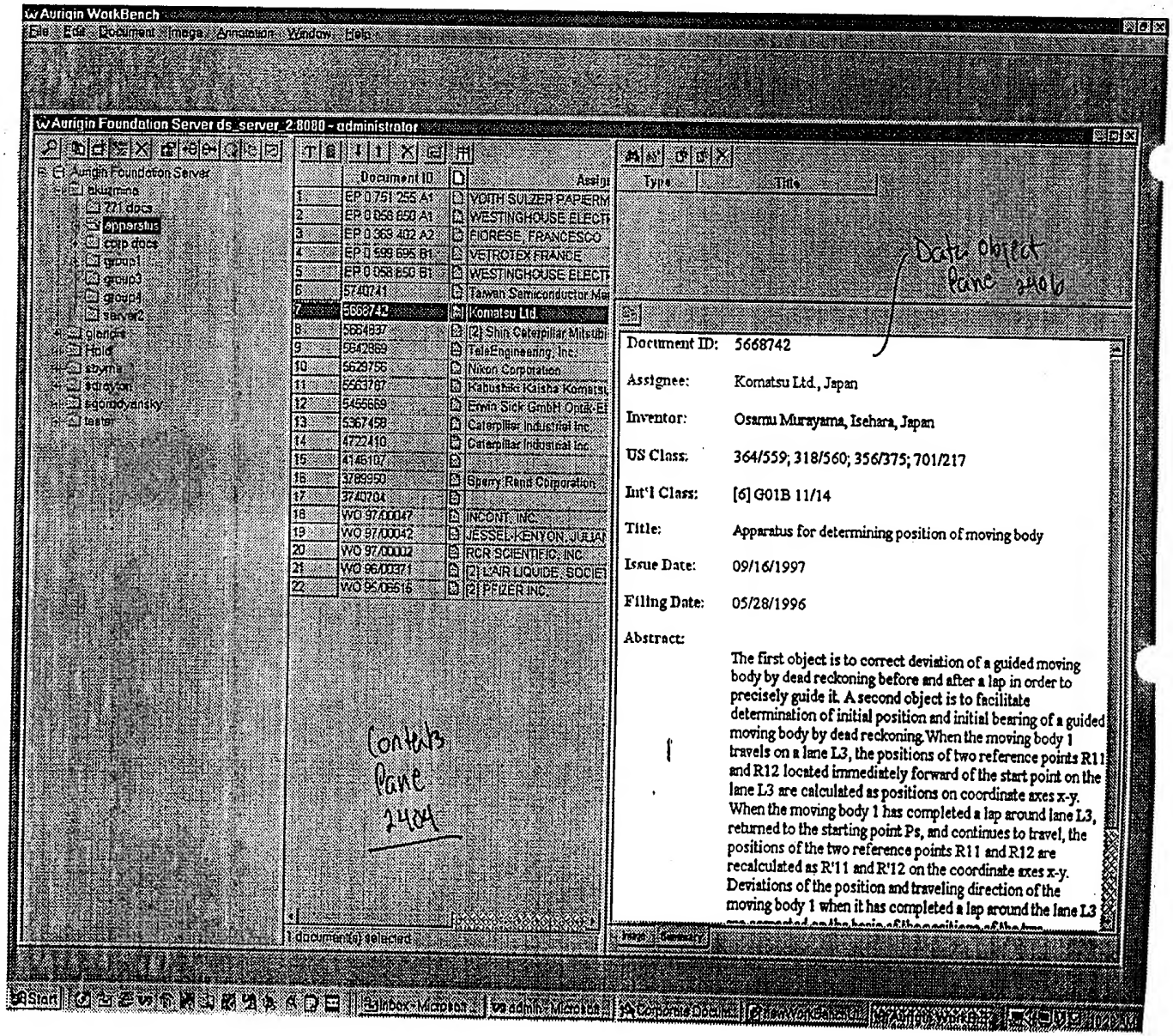


FIG. 24

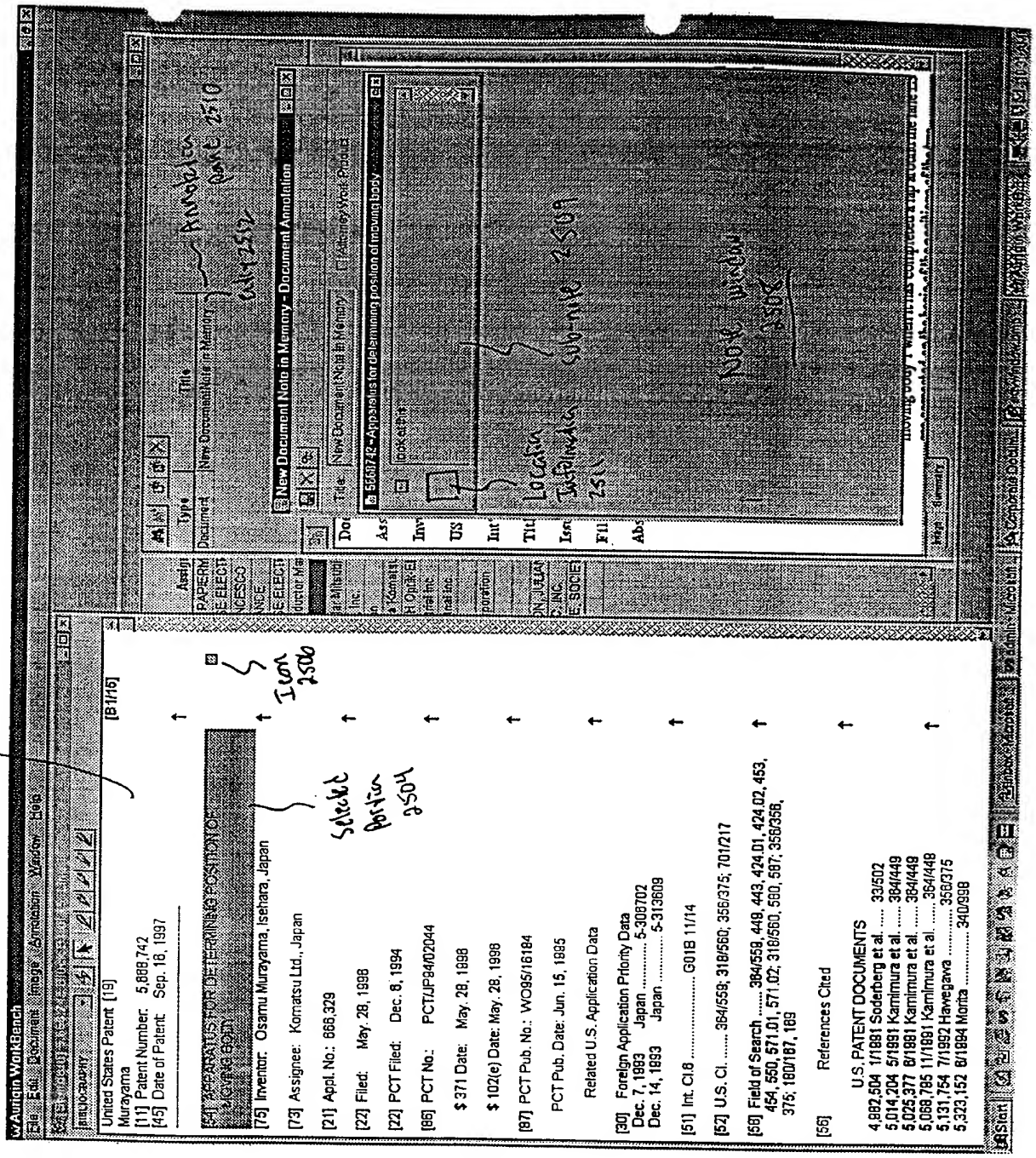
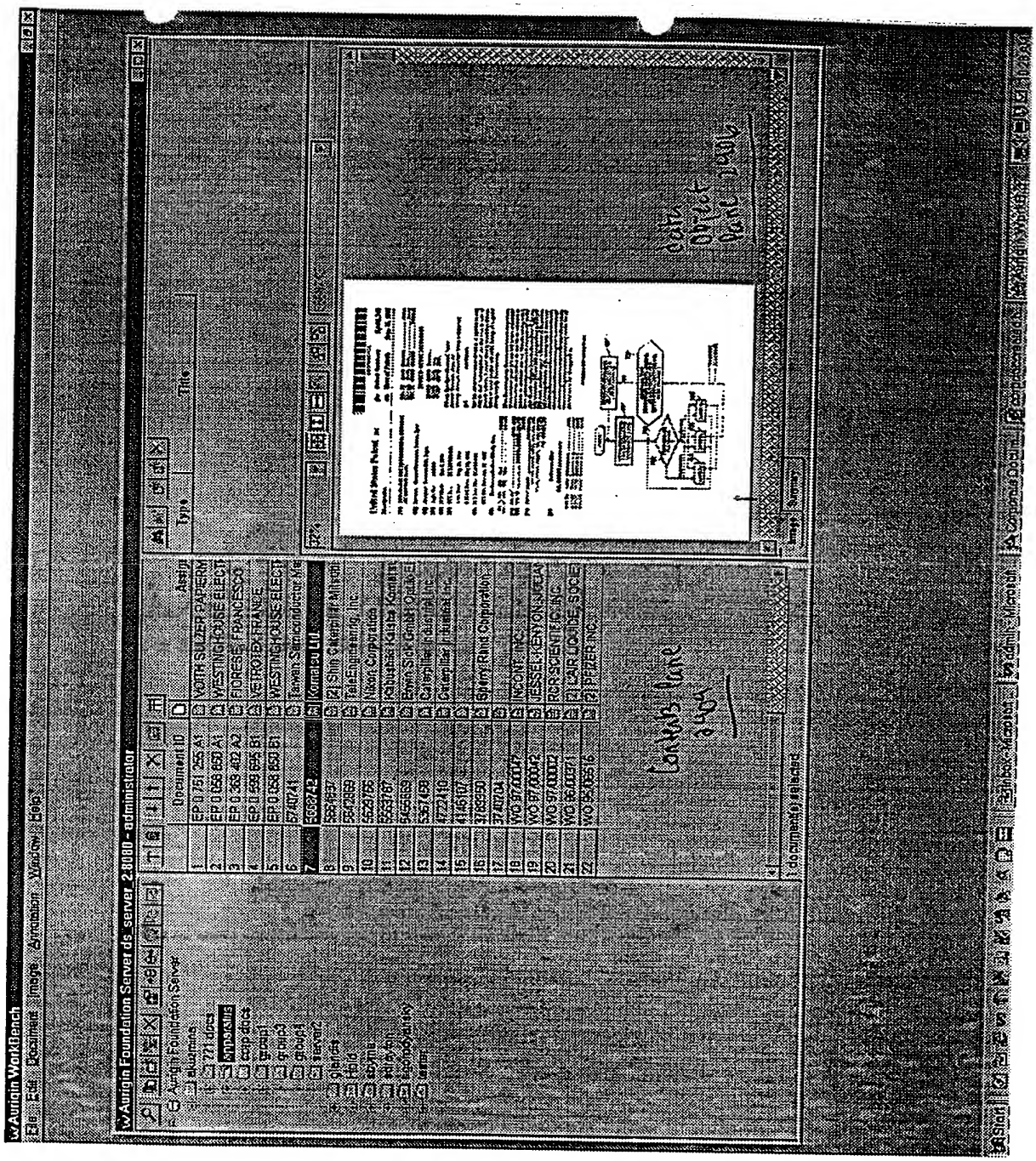

Browser 2402
Text window 2502
Icon 2508
Location 2510
Window 2504
Scroll bar 2506
Window 2504
Scroll bar 2506
AnyAuction
New Document Note in Memory - Document Annotation
[54] APPARATUS FOR DETERMINING POSITION OF MOVING BODY
[75] Inventor: Osamu Murayama, Isehara, Japan
[73] Assignee: Komatsu Ltd., Japan
[21] Appl. No.: 866,329
[22] Filed: May. 28, 1998
[22] PCT Filed: Dec. 8, 1994
[86] PCT No.: PCT/JP94/02044
§ 371 Date: May. 28, 1998
§ 102(e) Date: May. 28, 1998
[87] PCT Pub. No.: WO95/16184
PCT Pub. Date: Jun. 15, 1995
Related U.S. Application Data
[30] Foreign Application Priority Data
Dec. 7, 1993 Japan 5-308702
Dec. 14, 1993 Japan 5-313609
[51] Int. Cl.8 ..... G01B 11/14
[52] U.S. Cl. 364/559; 318/560; 356/375; 701/217
[58] Field of Search 364/559, 449, 443, 424.01, 424.02, 453, 454, 550, 571.01, 571.02; 318/560, 590, 587; 356/358, 375; 180/167, 169
[56] References Cited
U.S. PATENT DOCUMENTS
4,882,504 1/1881 Soderberg et al. 33/502
5,014,204 5/1991 Kamimura et al. 364/449
5,025,377 6/1991 Kamimura et al. 364/449
5,068,785 11/1991 Kamimura et al. 364/449
5,131,754 7/1992 Hawegawa 356/375
5,323,152 6/1994 Morita 340/998

FIG. 25


Browser #402
Document ID
Type
Title
EP 0 751 254 A1
EP 0 658 660 A1
EP 0 263 402 A2
EP 0 658 604 B1
EP 0 658 650 B1
57-40741
E0082A2
Komatsu Ltd.
Shin Caterpillar Mitsubishi
TelEngineering, Inc.
Nikon Corporation
Robotic Vision Systems
Emhart Glass Machinery
Caterpillar Industrial Inc.
Gerber Scientific, Inc.
Sperry Rand Corporation
Xerox Corporation
WO 97/00042
JESSE KENYON JUEGA
RICHARD SCHULTZ
CARLOMBE SOCIE
PFIZER INC.
WO 98/08816
Drawing Sheet 1 of 1
Browser #402
FIG. 26

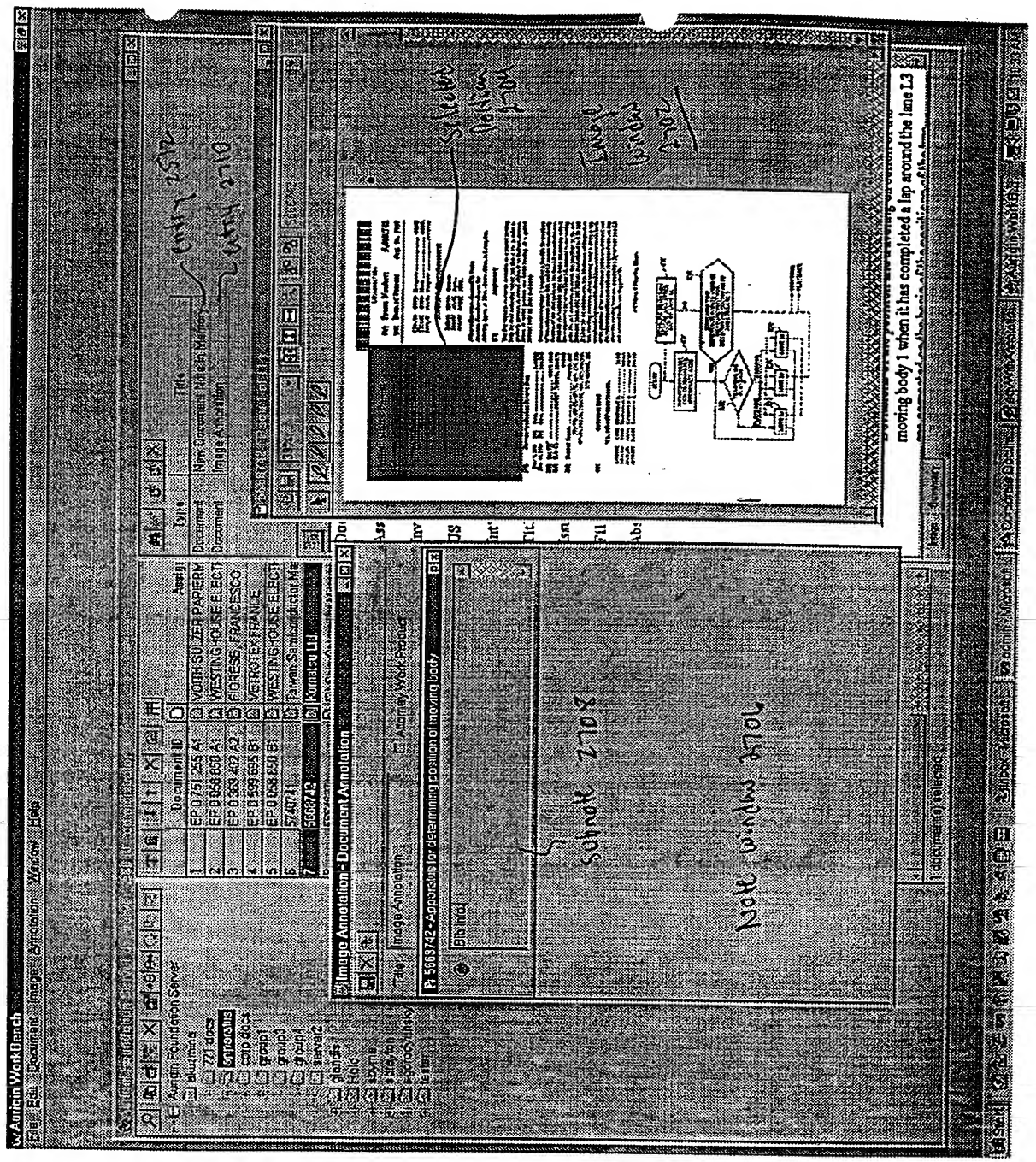


Fig. 27

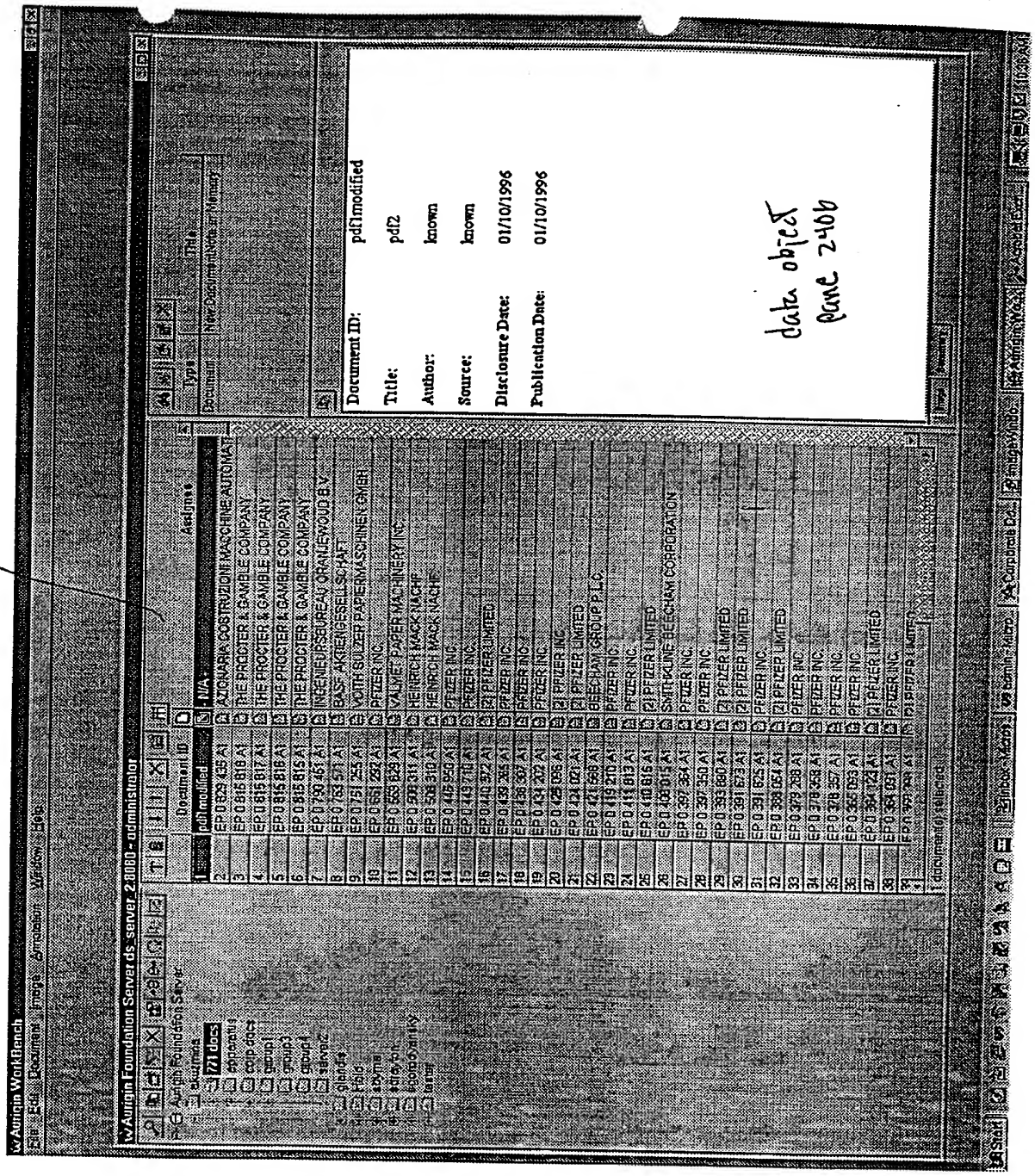

Document ID: pdf1modified
Title: pdf2
Author: known
Source: known
Disclosure Date: 01/10/1996
Publication Date: 01/10/1996
data object
page 2404

FIG. 28

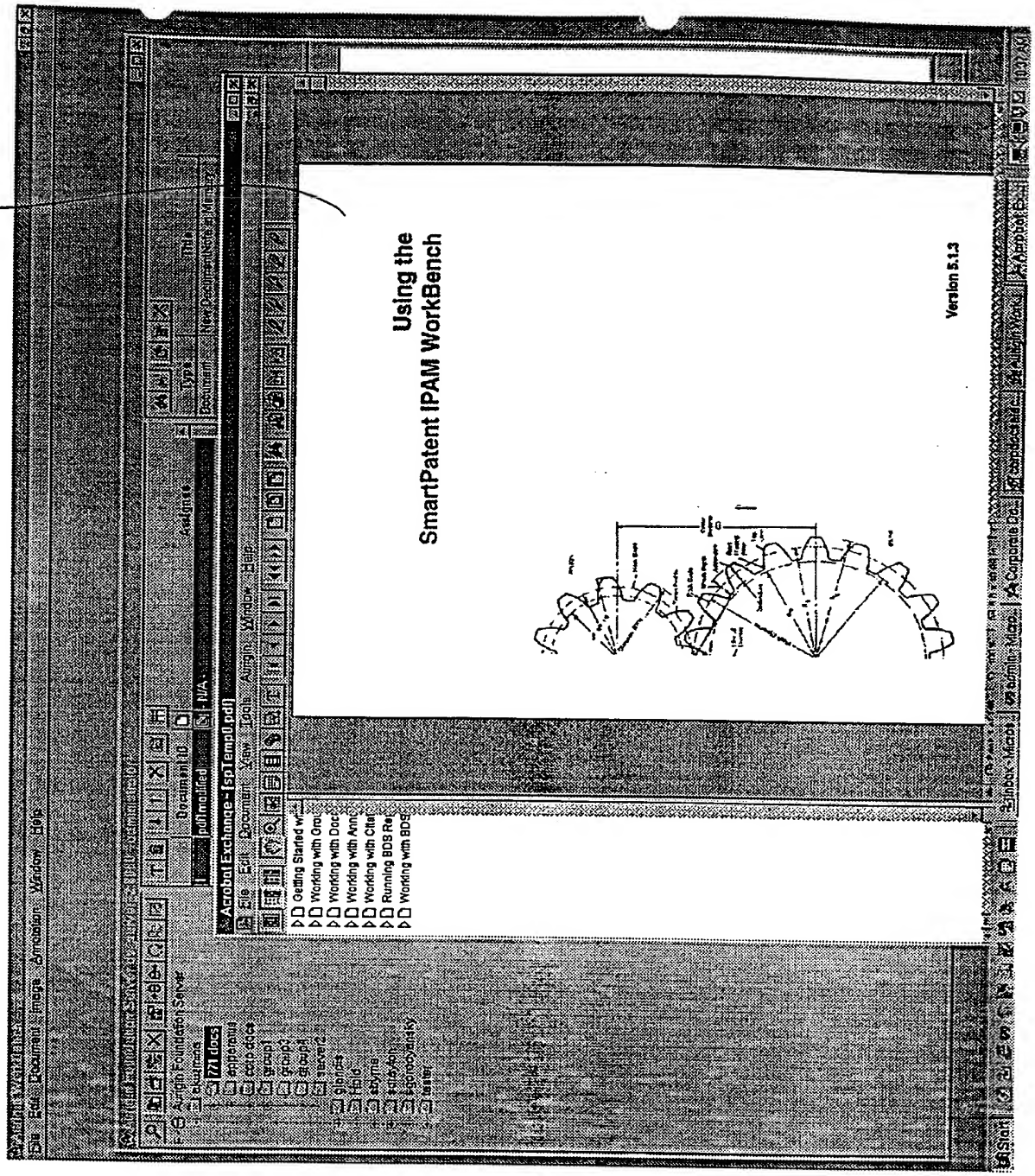


Image window
202
Using the
SmartPatent IPAM WorkBench
Version 5.1.3

FIG. 2g

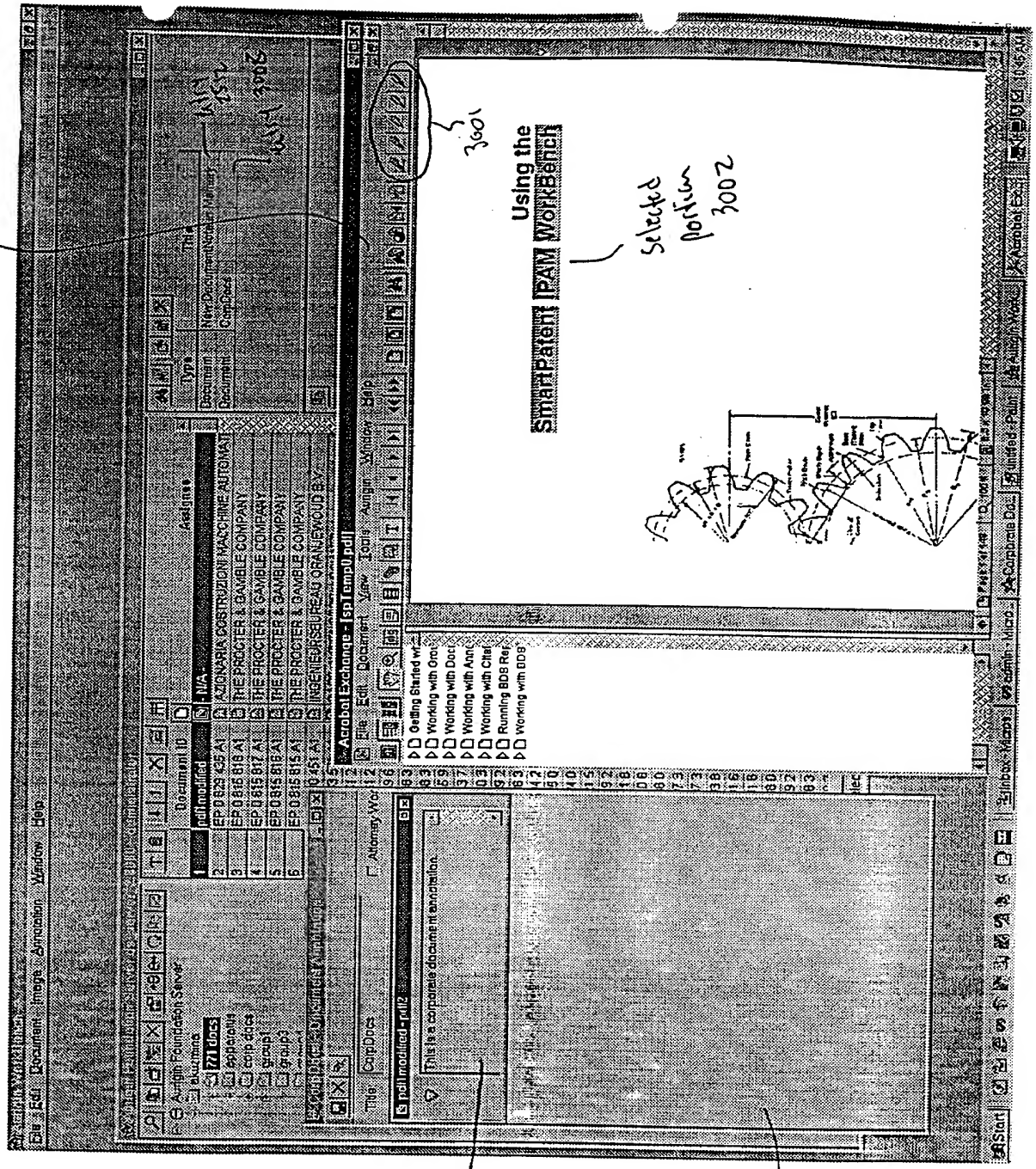

Using the
SmartPatent IPAM Workbench
Selected Portion 3002
3001
Image window 2902
Submit button 3006
Note window 3004

FIG. 30

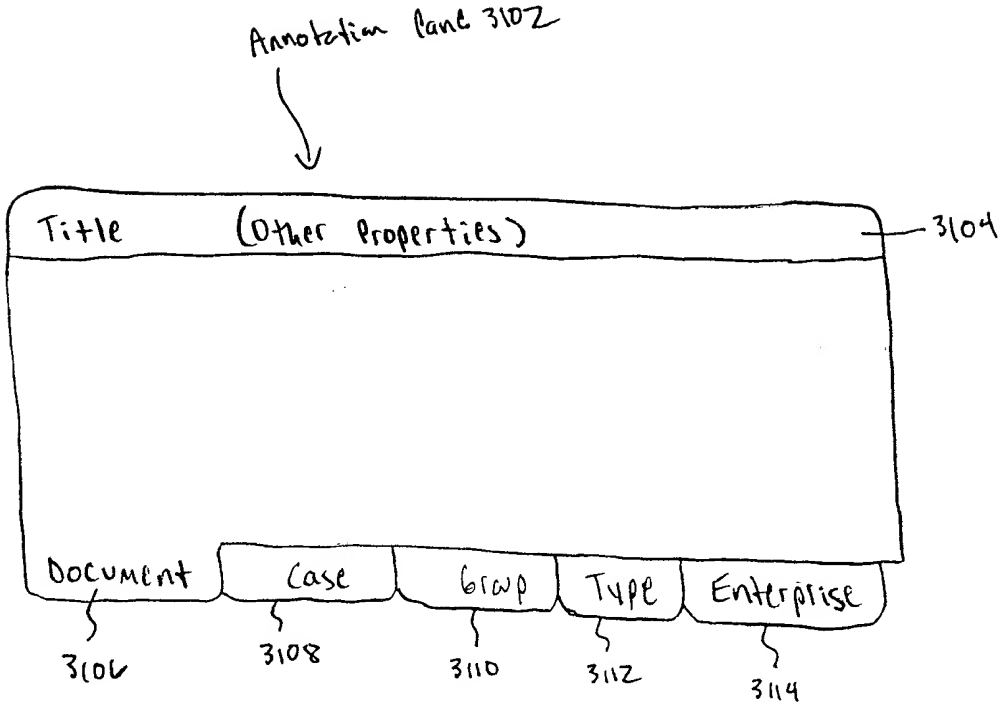

Annotation Panel 3102
Title
(Other Properties)
3104
Document
Case
Group
Type
Enterprise
3106
3108
3110
3112
3114

FIG. 31

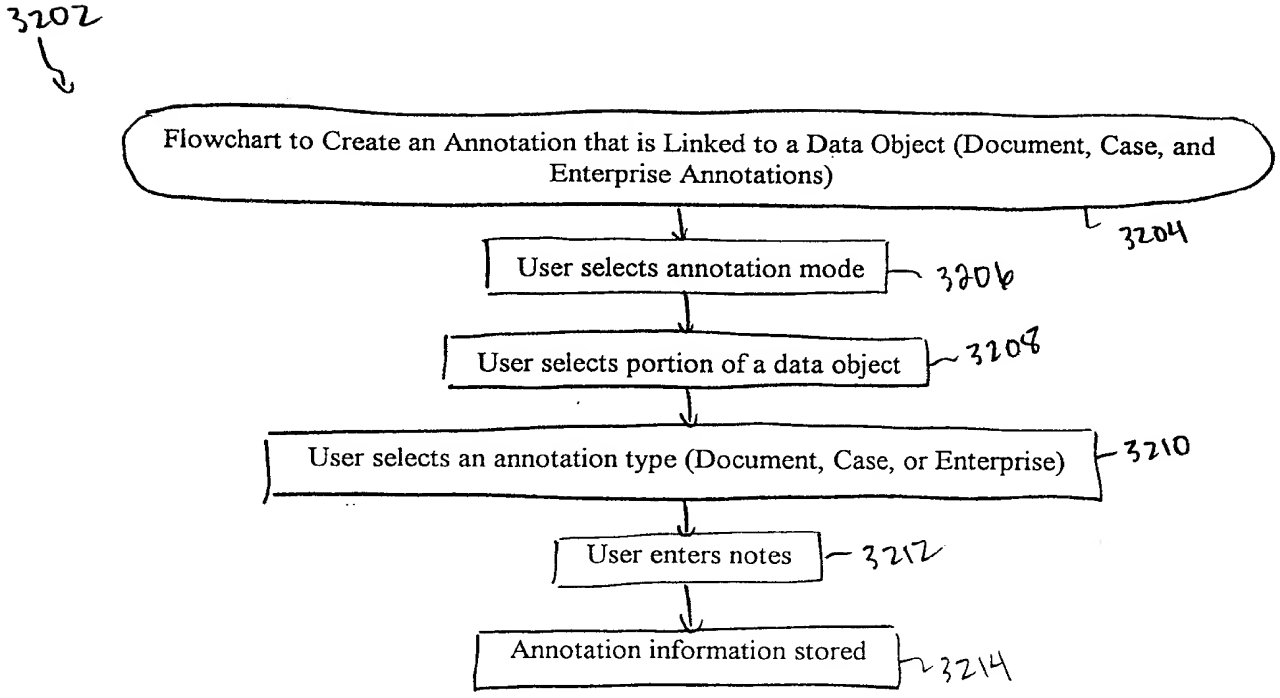

3202
Flowchart to Create an Annotation that is Linked to a Data Object (Document, Case, and Enterprise Annotations)
3204
User selects annotation mode
3206
User selects portion of a data object
3208
User selects an annotation type (Document, Case, or Enterprise)
3210
User enters notes
3212
Annotation information stored
3214

FIG. 32

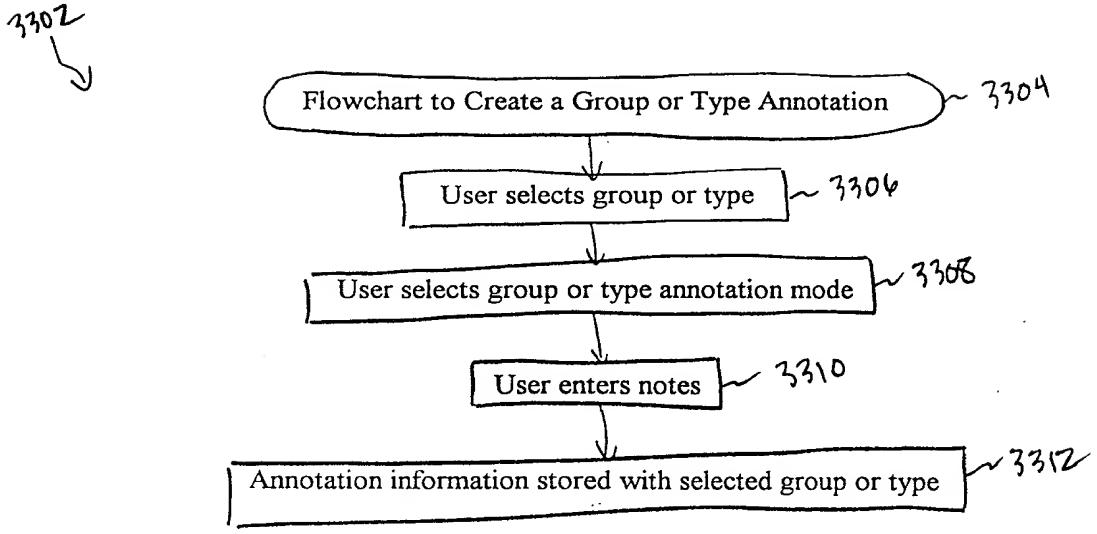

3302
Flowchart to Create a Group or Type Annotation
3304
User selects group or type
3306
User selects group or type annotation mode
3308
User enters notes
3310
Annotation information stored with selected group or type
3312

FIG. 33

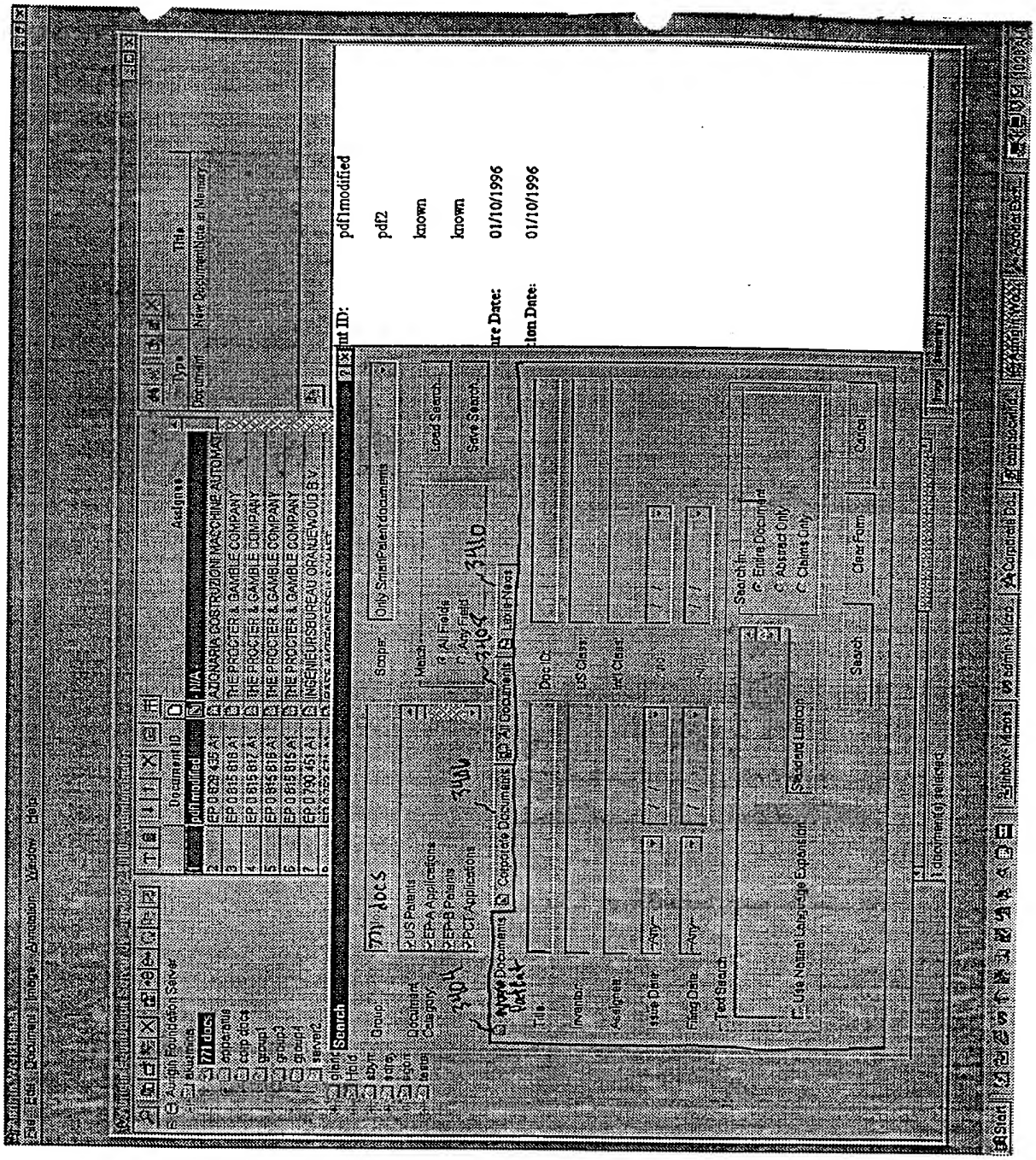

Search GUI 3402
3410
3412
3414
3416
3418
3420
3422
3424
3426
3428
3430
3432
3434
3436
3438
3440
3442
3444
3446
3448
3450
3452
3454
3456
3458
3460
3462
3464
3466
3468
3470
3472
3474
3476
3478
3480
3482
3484
3486
3488
3490
3492
3494
3496
3498
3500
3502
3504
3506
3508
3510
3512
3514
3516
3518
3520
3522
3524
3526
3528
3530
3532
3534
3536
3538
3540
3542
3544
3546
3548
3550
3552
3554
3556
3558
3560
3562
3564
3566
3568
3570
3572
3574
3576
3578
3580
3582
3584
3586
3588
3590
3592
3594
3596
3598
3600
3602
3604
3606
3608
3610
3612
3614
3616
3618
3620
3622
3624
3626
3628
3630
3632
3634
3636
3638
3640
3642
3644
3646
3648
3650
3652
3654
3656
3658
3660
3662
3664
3666
3668
3670
3672
3674
3676
3678
3680
3682
3684
3686
3688
3690
3692
3694
3696
3698
3700
3702
3704
3706
3708
3710
3712
3714
3716
3718
3720
3722
3724
3726
3728
3730
3732
3734
3736
3738
3740
3742
3744
3746
3748
3750
3752
3754
3756
3758
3760
3762
3764
3766
3768
3770
3772
3774
3776
3778
3780
3782
3784
3786
3788
3790
3792
3794
3796
3798
3800
3802
3804
3806
3808
3810
3812
3814
3816
3818
3820
3822
3824
3826
3828
3830
3832
3834
3836
3838
3840
3842
3844
3846
3848
3850
3852
3854
3856
3858
3860
3862
3864
3866
3868
3870
3872
3874
3876
3878
3880
3882
3884
3886
3888
3890
3892
3894
3896
3898
3900
3902
3904
3906
3908
3910
3912
3914
3916
3918
3920
3922
3924
3926
3928
3930
3932
3934
3936
3938
3940
3942
3944
3946
3948
3950
3952
3954
3956
3958
3960
3962
3964
3966
3968
3970
3972
3974
3976
3978
3980
3982
3984
3986
3988
3990
3992
3994
3996
3998
4000
4002
4004
4006
4008
4010
4012
4014
4016
4018
4020
4022
4024
4026
4028
4030
4032
4034
4036
4038
4040
4042
4044
4046
4048
4050
4052
4054
4056
4058
4060
4062
4064
4066
4068
4070
4072
4074
4076
4078
4080
4082
4084
4086
4088
4090
4092
4094
4096
4098
4100
4102
4104
4106
4108
4110
4112
4114
4116
4118
4120
4122
4124
4126
4128
4130
4132
4134
4136
4138
4140
4142
4144
4146
4148
4150
4152
4154
4156
4158
4160
4162
4164
4166
4168
4170
4172
4174
4176
4178
4180
4182
4184
4186
4188
4190
4192
4194
4196
4198
4200
4202
4204
4206
4208
4210
4212
4214
4216
4218
4220
4222
4224
4226
4228
4230
4232
4234
4236
4238
4240
4242
4244
4246
4248
4250
4252
4254
4256
4258
4260
4262
4264
4266
4268
4270
4272
4274
4276
4278
4280
4282
4284
4286
4288
4290
4292
4294
4296
4298
4300
4302
4304
4306
4308
4310
4312
4314
4316
4318
4320
4322
4324
4326
4328
4330
4332
4334
4336
4338
4340
4342
4344
4346
4348
4350
4352
4354
4356
4358
4360
4362
4364
4366
4368
4370
4372
4374
4376
4378
4380
4382
4384
4386
4388
4390
4392
4394
4396
4398
4400
4402
4404
4406
4408
4410
4412
4414
4416
4418
4420
4422
4424
4426
4428
4430
4432
4434
4436
4438
4440
4442
4444
4446
4448
4450
4452
4454
4456
4458
4460
4462
4464
4466
4468
4470
4472
4474
4476
4478
4480
4482
4484
4486
4488
4490
4492
4494
4496
4498
4500
4502
4504
4506
4508
4510
4512
4514
4516
4518
4520
4522
4524
4526
4528
4530
4532
4534
4536
4538
4540
4542
4544
4546
4548
4550
4552
4554
4556
4558
4560
4562
4564
4566
4568
4570
4572
4574
4576
4578
4580
4582
4584
4586
4588
4590
4592
4594
4596
4598
4600
4602
4604
4606
4608
4610
4612
4614
4616
4618
4620
4622
4624
4626
4628
4630
4632
4634
4636
4638
4640
4642
4644
4646
4648
4650
4652
4654
4656
4658
4660
4662
4664
4666
4668
4670
4672
4674
4676
4678
4680
4682
4684
4686
4688
4690
4692
4694
4696
4698
4700
4702
4704
4706
4708
4710
4712
4714
4716
4718
4720
4722
4724
4726
4728
4730
4732
4734
4736
4738
4740
4742
4744
4746
4748
4750
4752
4754
4756
4758
4760
4762
4764
4766
4768
4770
4772
4774
4776
4778
4780
4782
4784
4786
4788
4790
4792
4794
4796
4798
4800
4802
4804
4806
4808
4810
4812
4814
4816
4818
4820
4822
4824
4826
4828
4830
4832
4834
4836
4838
4840
4842
4844
4846
4848
4850
4852
4854
4856
4858
4860
4862
4864
4866
4868
4870
4872
4874
4876
4878
4880
4882
4884
4886
4888
4890
4892
4894
4896
4898
4900
4902
4904
4906
4908
4910
4912
4914
4916
4918
4920
4922
4924
4926
4928
4930
4932
4934
4936
4938
4940
4942
4944
4946
4948
4950
4952
4954
4956
4958
4960
4962
4964
4966
4968
4970
4972
4974
4976
4978
4980
4982
4984
4986
4988
4990
4992
4994
4996
4998
5000
5002
5004
5006
5008
5010
5012
5014
5016
5018
5020
5022
5024
5026
5028
5030
5032
5034
5036
5038
5040
5042
5044
5046
5048
5050
5052
5054
5056
5058
5060
5062
5064
5066
5068
5070
5072
5074
5076
5078
5080
5082
5084
5086
5088
5090
5092
5094
5096
5098
5100
5102
5104
5106
5108
5110
5112
5114
5116
5118
5120
5122
5124
5126
5128
5130
5132
5134
5136
5138
5140
5142
5144
5146
5148
5150
5152
5154
5156
5158
5160
5162
5164
5166
5168
5170
5172
5174
5176
5178
5180
5182
5184
5186
5188
5190
5192
5194
5196
5198
5200
5202
5204
5206
5208
5210
5212
5214
5216
5218
5220
5222
5224
5226
5228
5230
5232
5234
5236
5238
5240
5242
5244
5246
5248
5250
5252
5254
5256
5258
5260
5262
5264
5266
5268
5270
5272
5274
5276
5278
5280
5282
5284
5286
5288
5290
5292
5294
5296
5298
5300
5302
5304
5306
5308
5310
5312
5314
5316
5318
5320
5322
5324
5326
5328
5330
5332
5334
5336
5338
5340
5342
5344
5346
5348
5350
5352
5354
5356
5358
5360
5362
5364
5366
5368
5370
5372
5374
5376
5378
5380
5382
5384
5386
5388
5390
5392
5394
5396
5398
5400
5402
5404
5406
5408
5410
5412
5414
5416
5418
5420
5422
5424
5426
5428
5430
5432
5434
5436
5438
5440
5442
5444
5446
5448
5450
5452
5454
5456
5458
5460
5462
5464
5466
5468
5470
5472
5474
5476
5478
5480
5482
5484
5486
5488
5490
5492
5494
5496
5498
5500
5502
5504
5506
5508
5510
5512
5514
5516
5518
5520
5522
5524
5526
5528
5530
5532
5534
5536
5538
5540
5542
5544
5546
5548
5550
5552
5554
5556
5558
5560
5562
5564
5566
5568
5570
5572
5574
5576
5578
5580
5582
5584
5586
5588
5590
5592
5594
5596
5598
5600
5602
5604
5606
5608
5610
5612
5614
5616
5618
5620
5622
5624
5626
5628
5630
5632
5634
5636
5638
5640
5642
5644
5646
5648
5650
5652
5654
5656
5658
5660
5662
5664
5666
5668
5670
5672
5674
5676
5678
5680
5682
5684
5686
5688
5690
5692
5694
5696
5698
5700
5702
5704
5706
5708
5710
5712
5714
5716
5718
5720
5722
5724
5726
5728
5730
5732
5734
5736
5738
5740
5742
5744
5746
5748
5750
5752
5754
5756
5758
5760
5762
5764
5766
5768
5770
5772
5774
5776
5778
5780
5782
5784
5786
5788
5790
5792
5794
5796
5798
5800
5802
5804
5806
5808
5810
5812
5814
5816
5818
5820
5822
5824
5826
5828
5830
5832
5834
5836
5838
5840
5842
5844
5846
5848
5850
5852
5854
5856
5858
5860
5862
5864
5866
5868
5870
5872
5874
5876
5878
5880
5882
5884
5886
5888
5890
5892
5894
5896
5898
5900
5902
5904
5906
5908
5910
5912
5914
5916
5918
5920
5922
5924
5926
5928
5930
5932
5934
5936
5938
5940
5942
5944
5946
5948
5950
5952
5954
5956
5958
5960
5962
5964
5966
5968
5970
5972
5974
5976
5978
5980
5982
5984
5986
5988
5990
5992
5994
5996
5998
6000
6002
6004
6006
6008
6010
6012
6014
6016
6018
6020
6022
6024
6026
6028
6030
6032
6034
6036
6038
6040
6042
6044
6046
6048
6050
6052
6054
6056
6058
6060
6062
6064
6066
6068
6070
6072
6074
6076
6078
6080
6082
6084
6086
6088
6090
6092
6094
6096
6098
6100
6102
6104
6106
6108
6110
6112
6114
6116
6118
6120
6122
6124
6126
6128
6130
6132
6134
6136
6138
6140
6142
6144
6146
6148
6150
6152
6154
6156
6158
6160
6162
6164
6166
6168
6170
6172
6174
6176
6178
6180
6182
6184
6186
6188
6190
6192
6194
6196
6198
6200
6202
6204
6206
6208
6210
6212
6214
6216
6218
6220
6222
6224
6226
6228
6230
6232
6234
6236
6238
6240
6242
6244
6246
6248
6250
6252
6254
6256
6258
6260
6262
6264
6266
6268
6270
6272
6274
6276
6278
6280
6282
6284
6286
6288
6290
6292
6294
6296
6298
6300
6302
6304
6306
6308
6310
6312
6314
6316
6318
6320
6322
6324
6326
6328
6330
6332
6334
6336
6338
6340
6342
6344
6346
6348
6350
6352
6354
6356
6358
6360
6362
6364
6366
6368
6370
6372
6374
6376
6378
6380
6382
6384
6386
6388
6390
6392
6394
6396
6398
6400
6402
6404
6406
6408
6410
6412
6414
6416
6418
6420
6422
6424
6426
6428
6430
6432
6434
6436
6438
6440
6442
6444
6446
6448
6450
6452
6454
6456
6458
6460
6462
6464
6466
6468
6470
6472
6474
6476
6478
6480
6482
6484
6486
6488
6490
6492
6494
6496
6498
6500
6502
6504
6506
6508
6510
6512
6514
6516
6518
6520
6522
6524
6526
6528
6530
6532
6534
6536
6538
6540
6542
6544
6546
6548
6550
6552
6554
6556
6558
6560
6562
6564
6566
6568
6570
6572
6574
6576
6578
6580
6582
6584
6586
6588
6590
6592
6594
6596
6598
6600
6602
6604
6606
6608
6610
6612
6614
6616
6618
6620
6622
6624
6626
6628
6630
6632
6634
6636
6638
6640
6642
6644
6646
6648
6650
6652
6654
6656
6658
6660
6662
6664
6666
6668
6670
6672
6674
6676
6678
6680
6682
6684
6686
6688
6690
6692
6694
6696
6698
6700
6702
6704
6706
6708
6710
6712
6714
6716
6718
6720
6722
6724
6726
6728
6730
6732
6734
6736
6738
6740
6742
6744
6746
6748
6750
6752
6754
6756
6758
6760
6762
6764
6766
6768
6770
6772
6774
6776
6778
6780
6782
6784
6786
6788
6790
6792
6794
6796
6798
6800
6802
6804
6806
6808
6810
6812
6814
6816
6818
6820
6822
6824
6826
6828
6830
6832
6834
6836
6838
6840
6842
6844
6846
6848
6850
6852
6854
6856
6858
6860
6862
6864
6866
6868
6870
6872
6874
6876
6878
6880
6882
6884
6886
6888
6890
6892
6894
6896
6898
6900
6902
6904
6906
6908
6910
6912
6914
6916
6918
6920
6922
6924
6926
6928
6930
6932
6934
6936
6938
6940
6942
6944
6946
6948
6950
6952
6954
6956
6958
6960
6962
6964
6966
6968
6970
6972
6974
6976
6978
6980
6982
6984
6986
6988
6990
6992
6994
6996
6998
7000
7002
7004
7006
7008
7010
7012
7014
7016
7018
7020
7022
7024
7026
7028
7030
7032
7034
7036
7038
7040
7042
7044
7046
7048
7050
7052
7054
7056
7058
7060
7062
7064
7066
7068
7070
7072
7074
7076
7078
7080
7082
7084
7086
7088
7090
7092
7094
7096
7098
7100
7102
7104
7106
7108
7110
7112
7114
7116
7118
7120
7122
7124
7126
7128
7130
7132
7134
7136
7138
7140
7142
7144
7146
7148
7150
7152
7154
7156
7158
7160
7162
7164
7166
7168
7170
7172
7174
7176
7178
7180
7182
7184
7186
7188
7190
7192
7194
7196
7198
7200
7202
7204
7206
7208
7210
7212
7214
7216
7218
7220
7222
7224
7226
7228
7230
7232
7234
7236
7238
7240
7242
7244
7246
7248
7250
7252
7254
7256
7258
7260
7262
7264
7266
7268
7270
7272
7274
7276
7278
7280
7282
7284
7286
7288
7290
7292
7294
7296
7298
7300
7302
7304
7306
7308
7310
7312
7314
7316
7318
7320
7322
7324
7326
7328
7330
7332
7334
7336
7338
7340
7342
7344
7346
7348
7350
7352
7354
7356
7358
7360
7362
7364
7366
7368
7370
7372
7374
7376
7378
7380
7382
7384
7386
7388
7390
7392
7394
7396
7398
7400
7402
7404
7406
7408
7410
7412
7414
7416
7418
7420
7422
7424
7426
7428
7430
7432
7434
7436
7438
7440
7442
7444
7446
7448
7450
7452
7454
7456
7458
7460
7462
7464
7466
7468
7470
7472
7474
7476
7478
7480
7482
7484
7486
7488
7490
7492
7494
7496
7498
7500
7502
7504
7506
7508
7510
7512
7514
7516
7518
7520
7522
7524
7526
7528
7530
7532
7534
7536
7538
7540
7542
7544
7546
7548
7550
7552
7554
7556
7558
7560
7562
7564
7566
7568
7570
7572
7574
7576
7578
7580
7582
7584
7586
7588
7590
7592
7594
7596
7598
7600
7602
7604
7606
7608
7610
7612
7614
7616
7618
7620
7622
7624
7626
7628
7630
7632
7634
7636
7638
7640
7642
7644
7646
7648
7650
7652
7654
7656
7658
7660
7662
7664
7666
7668
7670
7672
7674
7676
7678
7680
7682
7684
7686
7688
7690
7692
7694
7696
7698
7700
7702
7704
7706
7708
7710
7712
7714
7716
7718
7720
7722
7724
7726
7728
7730
7732
7734
7736
7738
7740
7742
7744
7746
7748
7750
7752
7754
7756
7758
7760
7762
7764
7766
7768
7770
7772
7774
7776
7778
7780
7782
7784
7786
7788
7790
7792
7794
7796
7798
7800
7802
7804
7806
7808
7810
7812
7814
7816
7818
7820
7822
7824
7826
7828
7830
7832
7834
7836
7838
7840
7842
7844
7846
7848
7850
7852
7854
7856
7858
7860
7862
7864
7866
7868
7870
7872
7874
7876
7878
7880
7882
7884
7886
7888
7890
7892
7894
7896
7898
7900
7902
7904
7906
7908
7910
7912
7914
7916
7918
7920
7922
7924
7926
7928
7930
7932
7934
7936
7938
7940
7942
7944
7946
7948
7950
7952
7954
7956
7958
7960
7962
7964
7966
7968
7970
7972
7974
7976
7978
7980
7982
7984
7986
7988
7990
7992
7994
7996
7998
8000
8002
8004
8006
8008
8010
8012
8014
8016
8018
8020
8022
8024
8026
8028
8030
8032
8034
8036
8038
8040
8042
8044
8046
8048
8050
8052
8054
8056
8058
8060
8062
8064
8066
8068
8070
8072
8074
8076
8078
8080
8082
8084
8086
8088
8090
8092
8094
8096
8098
8100
8102
8104
8106
8108
8110
8112
8114
8116
8118
8120
8122
8124
8126
8128
8130
8132
8134
8136
8138
8140
8142
8144
8146
8148
8150
8152
8154
8156
8158
8160
8162
8164
8166
8168
8170
8172
8174
8176
8178
8180
8182
8184
8186
8188
8190
8192
8194
8196
8198
8200
8202
8204
8206
8208
8210
8212
8214
8216
8218
8220
8222
8224
8226
8228
8230
8232
8234
8236
8238
8240
8242
8244
8246
8248
8250
8252
8254
8256
8258
8260
8262
8264
8266
8268
8270
8272
8274
8276
8278
8280
8282
8284
8286
8288
8290
8292
8294
8296
8298
8300
8302
8304
8306
8308
8310
8312
8314
8316
8318
8320
8322
8324
8326
8328
8330
8332
8334
8336
8338
8340
8342
8344
8346
8348
8350
8352
8354
8356
8358
8360
8362
8364
8366
8368
8370
8372
8374
8376
8378
8380
8382
8384
8386
8388
8390
8392
8394
8396
8398
8400
8402
8404
8406
8408
8410
8412
8414
8416
8418
8420
8422
8424
8426
8428
8430
8432
8434
8436
8438
8440
8442
8444
8446
8448
8450
8452
8454
8456
8458
8460
8462
8464
8466
8468
8470
8472
8474
8476
8478
8480
8482
8484
8486
8488
8490
8492
8494
8496
8498
8500
8502
8504
8506
8508
8510
8512
8514
8516
8518
8520
8522
8524
8526
8528
8530
8532
8534
8536
8538
8540
8542
8544
8546
8548
8550
8552
8554
8556
8558
8560
8562
8564
8566
8568
8570
8572
8574
8576
8578
8580
8582
8584
8586
8588
8590
8592
8594
8596
8598
8600
8602
8604
8606
8608
8610
8612
8614
8616
8618
8620
8622
8624
8626
8628
8630
8632
8634
8636
8638
8640
8642
8644
8646
8648
8650
8652
8654
8656
8658
8660
8662
8664
8666
8668
8670
8672
8674
8676
8678
8680
8682
8684
8686
8688
8690
8692
8694
8696
8698
8700
8702
8704
8706
8708
8710
8712
8714
8716
8718
8720
8722
8724
8726
8728
8730
8732
8734
8736
8738
8740
8742
8744
8746
8748
8750
8752
8754
8756
8758
8760
8762
8764
8766
8768
8770
8772
8774
8776
8778
8780
8782
8784
8786
8788
8790
8792
8794
8796
8798
8800
8802
8804
8806
8808
8810
8812
8814
8816
8818
8820
8822
8824
8826
8828
8830
8832
8834
8836
8838
8840
8842
8844
8846
8848
8850
8852
8854
8856
8858
8860
8862
8864
8866
8868
8870
8872
8874
8876
8878
8880
8882
8884
8886
8888
8890
8892
8894
8896
8898
8900
8902
8904
8906
8908
8910
8912
8914
8916
8918
8920
8922
8924
8926
8928
8930
8932
8934
8936
8938
8940
8942
8944
8946
8948
8950
8952
8954
8956
8958
8960
8962
8964
8966
8968
8970
8972
8974
8976
8978
8980
8982
8984
8986
8988
8990
8992
8994
8996
8998
9000
9002
9004
9006
9008
9010
9012
9014
9016
9018
9020
9022
9024
9026
9028
9030
9032
9034
9036
9038
9040
9042
9044
9046
9048
9050
9052
9054
9056
9058
9060
9062
9064
9066
9068
9070
9072
9074
9076
9078
9080
9082
9084
9086
9088
9090
9092
9094
9096
9098
9100
9102
9104
9106
9108
9110
9112
9114
9116
9118
9120
9122
9124
9126
9128
9130
9132
9134
9136
9138
9140
9142
9144
9146
9148
9150
9152
9154
9156
9158
9160
9162
9164
9166
9168
9170
9172
9174
9176
9178
9180
9182
9184
9186
9188
9190
9192
9194
9196
9198
9200
9202
9204
9206
9208
9210
9212
9214
9216
9218
9220
9222
9224
9226
9228
9230
9232
9234
9236
9238
9240
9242
9244
9246
9248
9250
9252
9254
9256
9258
9260
9262
9264
9266
9268
9270
9272
9274
9276
9278
9280
9282
9284
9286
9288
9290
9292
9294
9296
9298
9300
9302
9304
9306
9308
9310
9312
9314
9316
9318
9320
9322
9324
9326
9328
9330
9332
9334
9336
9338
9340
9342
9344
9346
9348
9350
9352
9354
9356
9358
9360
9362
9364
9366
9368
9370
9372
9374
9376
9378
9380
9382
9384
9386
9388
9390
9392
9394
9396
9398
9400
9402
9404
9406
9408
9410
9412
9414
9416
9418
9420
9422
9424
9426
9428
9430
9432
9434
9436
9438
9440
9442
9444
9446
9448
9450
9452
9454
9456
9458
9460
9462
9464
9466
9468
9470
9472
9474
9476
9478
9480
9482
9484
9486
9488
9490
9492
9494
9496
9498
9500
9502
9504
9506
9508
9510
9512
9514
9516
9518
9520
9522
9524
9526
9528
9530
9532
9534
9536
9538
9540
9542
9544
9546
9548
9550
9552
9554
9556
9558
9560
9562
9564
9566
9568
9570
9572
9574
9576
9578
9580
9582
9584
9586
9588
9590
9592
9594
9596
9598
9600
9602
9604
9606
9608
9610
9612
9614
9616
9618
9620
9622
9624
9626
9628
9630
9632
9634
9636
9638
9640
9642
9644
9646
9648
9650
9652
9654
9656
9658
9660
9662
9664
9666
9668
9670
9672
9674
9676
9678
9680
9682
9684
9686
9688
9690
9692
9694
9696
9698
9700
9702
9704
9706
9708
9710
9712
9714
9716
9718
9720
9722
9724
9726
9728
9730
9732
9734
9736
9738
9740
9742
9744
9746
9748
9750
9752
9754
9756
9758
9760
9762
9764
9766
9768
9770
9772
9774
9776
9778
9780
9782
9784
9786
9788
9790
9792
9794
9796
9798
9800
9802
9804
9806
9808
9810
9812
9814
9816
9818
9820
9822
9824
9826
9828
9830
9832
9834
9836
9838
9840
9842
9844
9846
9848
9850
9852
9854
9856
9858
9860
9862
9864
9866
9868
9870
9872
9874
9876
9878
9880
9882
9884
9886
9888
9890
9892
9894
9896
9898
9900
9902
9904
9906
9908
9910
9912
9914
9916
9918
9920
9922
9924
9926
9928
9930
9932
9934
9936
9938
9940
9942
9944
9946
9948
9950
9952
9954
9956
9958
9960
9962
9964
9966
9968
9970
9972
9974
9976
9978
9980
9982
9984
9986
9988
9990
9992
9994
9996
9998
10000

FIG. 34

Search GUI 3402
3404
3406
3408
3410
pdf1modified
pdf2
known
known
01/10/1996
01/10/1996
Search
Clear Form
Cancel
Load Search
Save Search
All Documents
Current Documents
Use Natural Language Expansion
Standard Language

FIG. 35

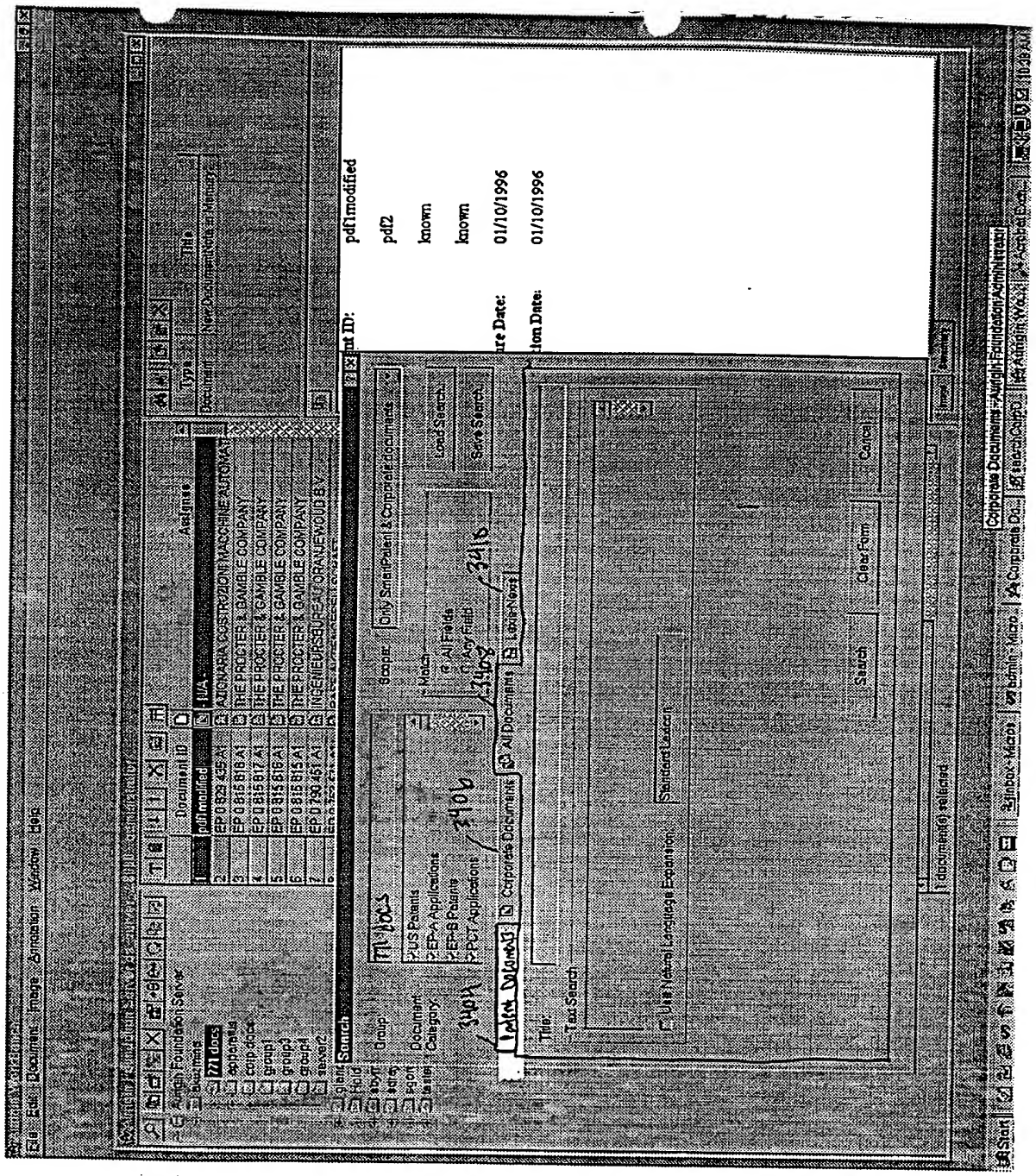
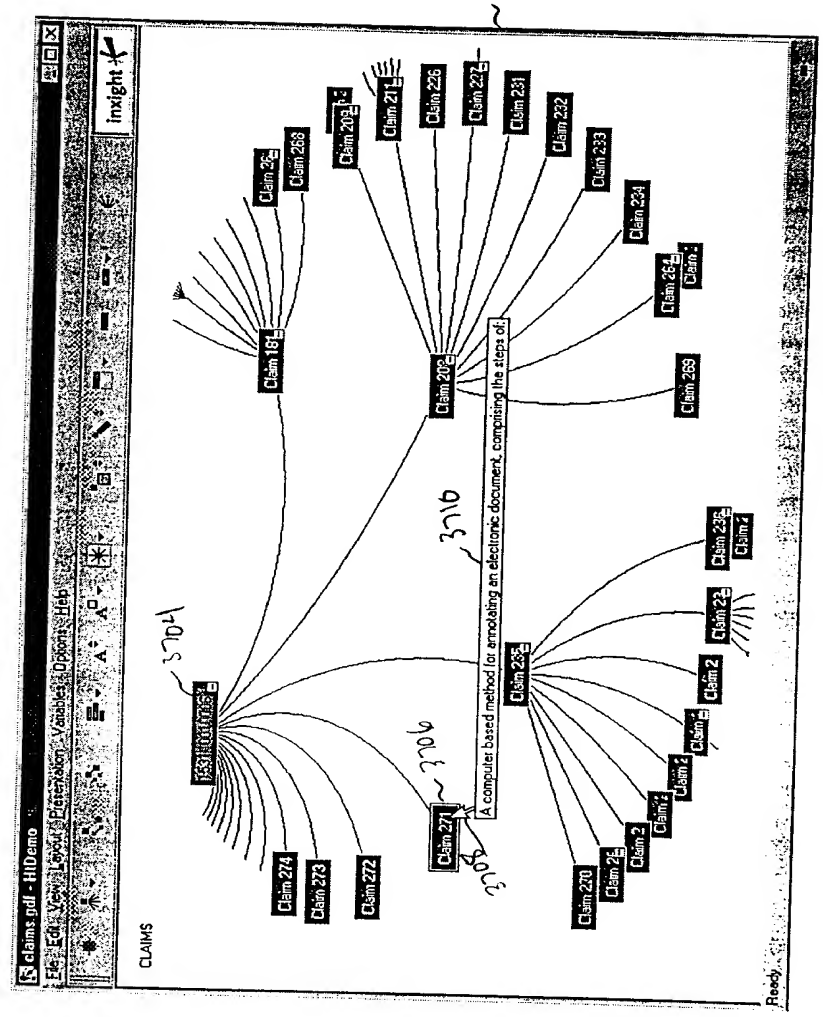

Search GUI
3402
3404
3406
3408
3410
3412
3414
3416
3418
pdf1modified
pdf2
known
known
01/10/1996
01/10/1996

FIG. 34



window 3712

3704

3710

3706

3708

Claim Tree 3702

FIG. 37

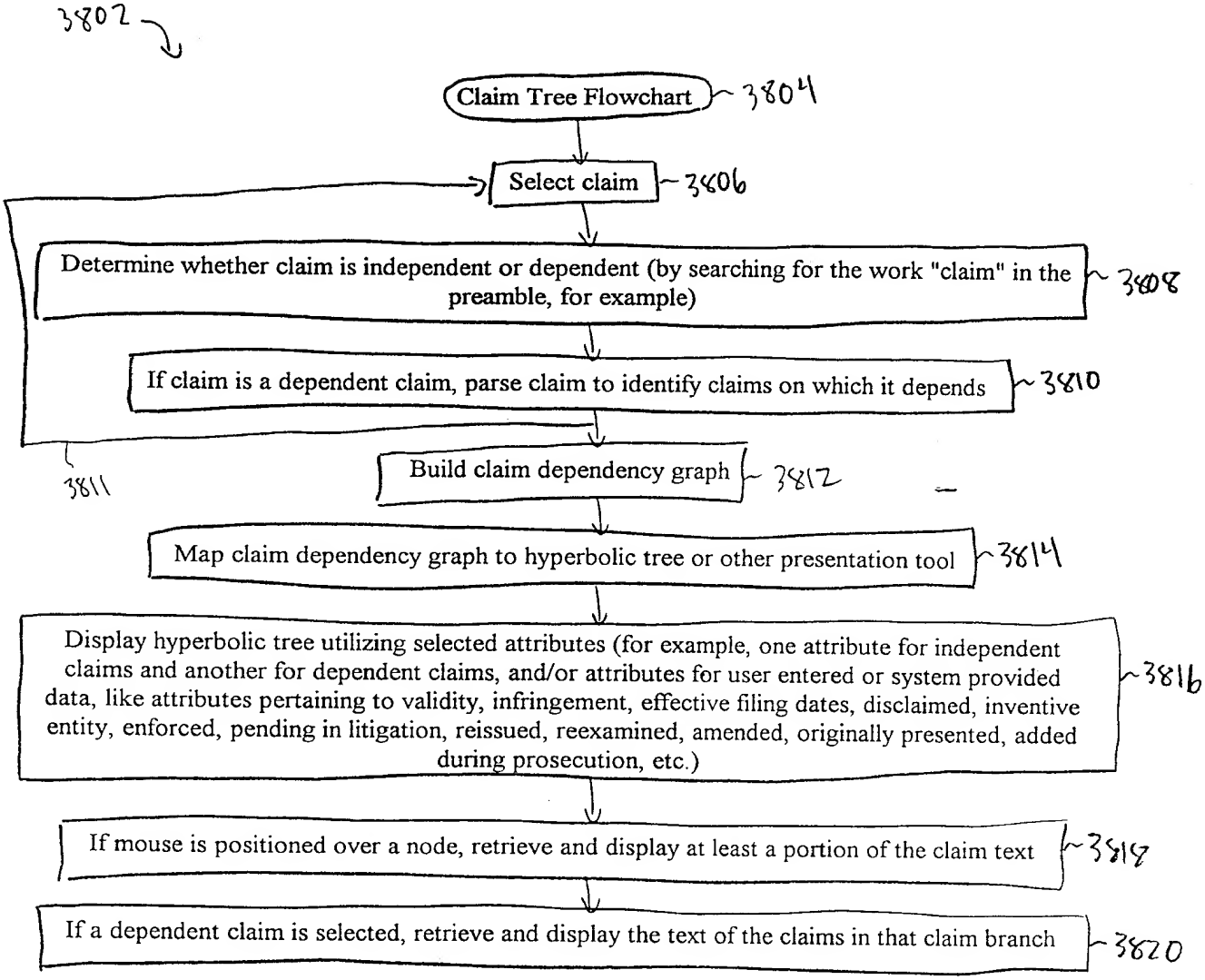

3802
Claim Tree Flowchart
3804
Select claim
3806
Determine whether claim is independent or dependent (by searching for the work "claim" in the preamble, for example)
3808
If claim is a dependent claim, parse claim to identify claims on which it depends
3810
3811
Build claim dependency graph
3812
Map claim dependency graph to hyperbolic tree or other presentation tool
3814
Display hyperbolic tree utilizing selected attributes (for example, one attribute for independent claims and another for dependent claims, and/or attributes for user entered or system provided data, like attributes pertaining to validity, infringement, effective filing dates, disclaimed, inventive entity, enforced, pending in litigation, reissued, reexamined, amended, originally presented, added during prosecution, etc.)
3816
If mouse is positioned over a node, retrieve and display at least a portion of the claim text
3818
If a dependent claim is selected, retrieve and display the text of the claims in that claim branch
3820

FIG. 38

235. A computer program product, comprising a computer useable medium having computer program logic stored therein, wherein said computer program logic enables a computer to annotate an electronic document, said computer program logic comprising:

document displaying means for enabling the computer to display at least a part of said electronic document;

note displaying means for enabling the computer to display at least one note having at least one note segment linked to at least one portion of said at least a part of said electronic document;

location identification information displaying means for enabling the computer to display in said at least one note, proximate to said at least one note segment, location identification information identifying a location of said at least one portion in said at least a part of said electronic document; and

means for enabling the computer to enable a user to create a new note segment in said at least one note, said new note segment being linked to at least a portion of said at least one portion of said at least a part of said electronic document;

wherein said new note segment and said at least one note segment are nested note segments.

3902

FIG. 39

270. The computer program product of claim 235, wherein said computer program logic further comprises:

means for enabling the computer to enable a user to insert into said at least one note segment at least one of

(a) a user-selected portion of said electronic document,
(b) a user-selected portion from another electronic document, and
(c) user-created work product.

FIG. 40

4602

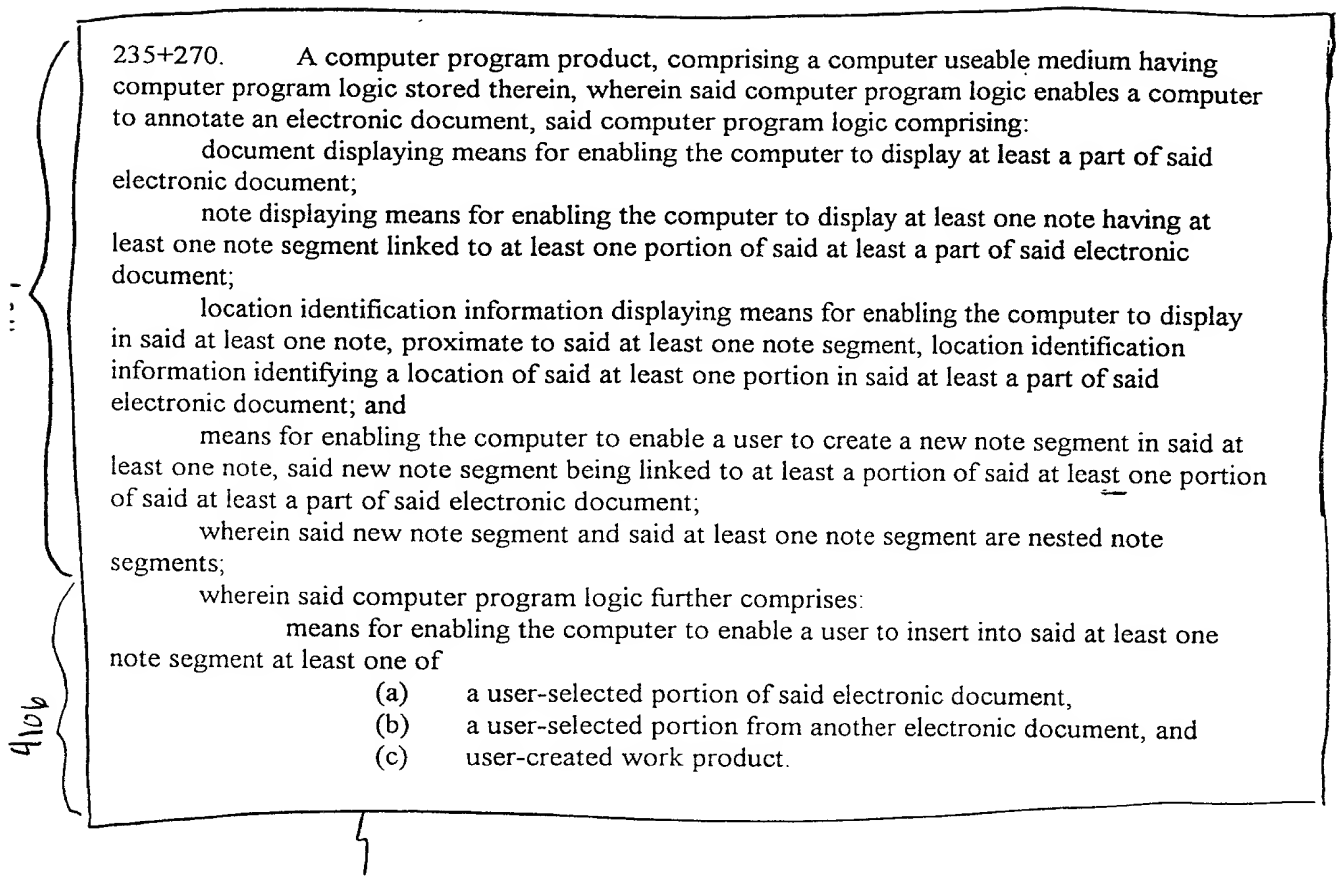

235+270. A computer program product, comprising a computer useable medium having computer program logic stored therein, wherein said computer program logic enables a computer to annotate an electronic document, said computer program logic comprising:
document displaying means for enabling the computer to display at least a part of said electronic document;
note displaying means for enabling the computer to display at least one note having at least one note segment linked to at least one portion of said at least a part of said electronic document;
location identification information displaying means for enabling the computer to display in said at least one note, proximate to said at least one note segment, location identification information identifying a location of said at least one portion in said at least a part of said electronic document; and
means for enabling the computer to enable a user to create a new note segment in said at least one note, said new note segment being linked to at least a portion of said at least one portion of said at least a part of said electronic document;
wherein said new note segment and said at least one note segment are nested note segments;
wherein said computer program logic further comprises:
means for enabling the computer to enable a user to insert into said at least one note segment at least one of
(a) a user-selected portion of said electronic document,
(b) a user-selected portion from another electronic document, and
(c) user-created work product.
4106
4107

FIG. 41

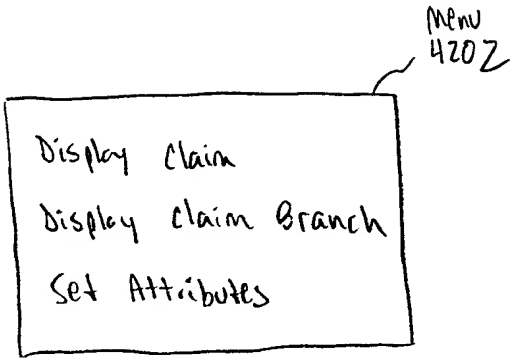

Menu
420Z
Display Claim
Display Claim Branch
Set Attributes

FIG 42A

Node State Information 4202
Label
Link Information
Display Attributes
font
color
bold
italics
underlining
opaque | transparent
visible | invisible
user defined

FIG. 42B

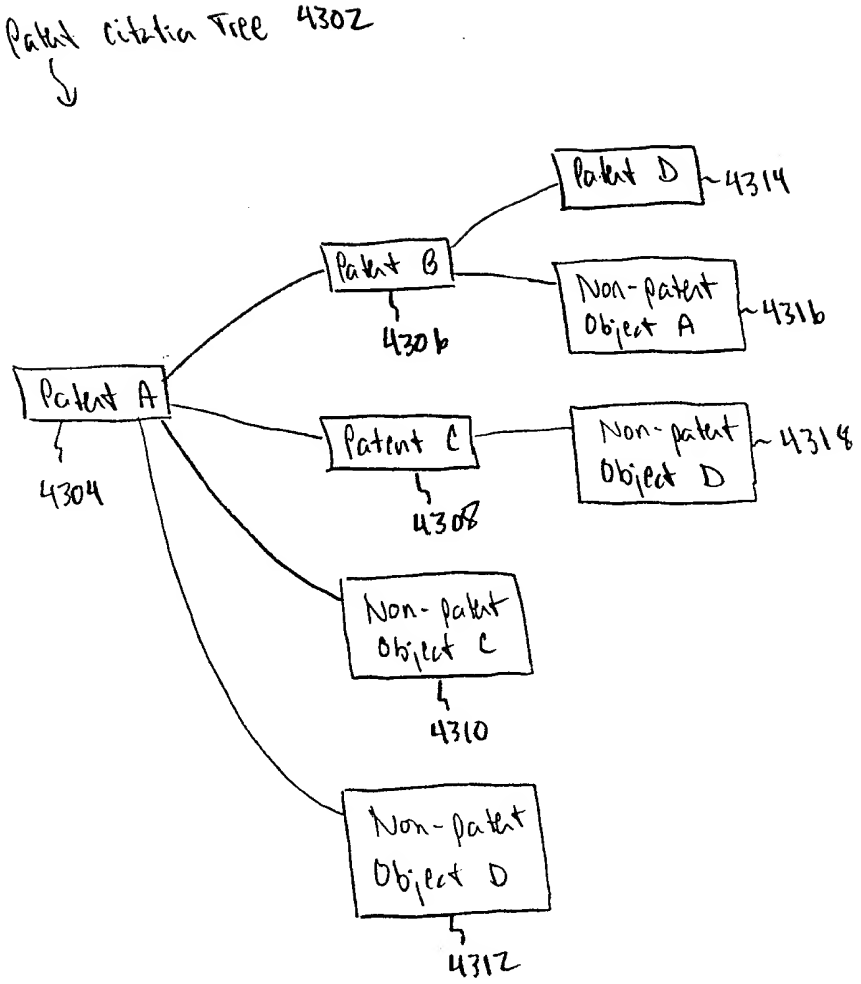

Patent citation Tree 4302
Patent D
4314
Patent B
4306
Non-patent Object A
4316
Patent A
4304
Patent C
4308
Non-patent Object D
4318
Non-patent Object C
4310
Non-patent Object D
4312

FIG. 43

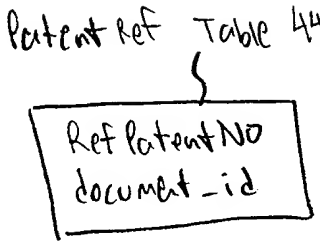


FIG. 44

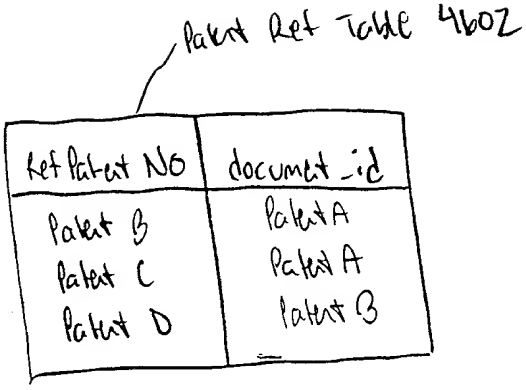


FIG. 46

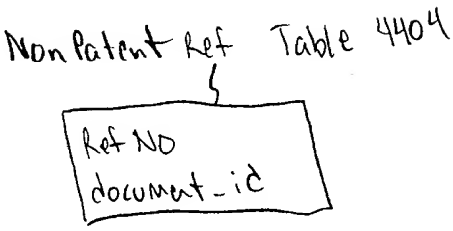


FIG. 45

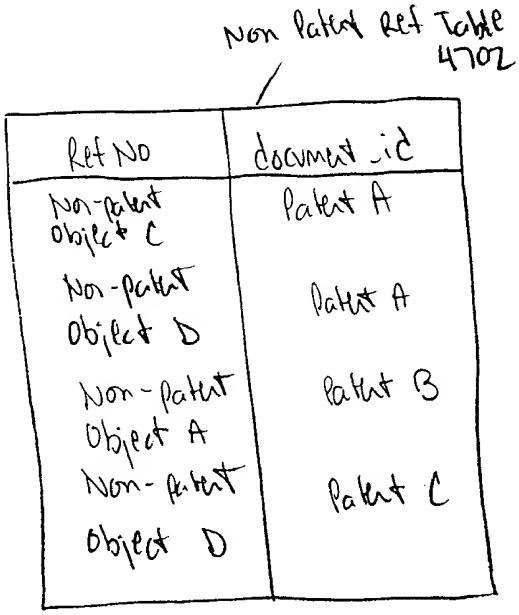


FIG. 47

Data Object Family Table 4802

From (4804)	To (4806)	Relationship Type (4808)

FIG. 48

Data Object Family Table 4902

From	To	Relationship Type	
Patent A	Patent B	Backward citation	4904
Patent A	Patent C	Backward citation	4906
Patent A	Non-Patent Object C	Backward citation	4908
Patent A	Non-Patent Object D	Backward citation	4910
Patent B	Patent D	Backward citation	4912
Patent B	Non-Patent Object A	Backward citation	4914
Patent C	Non-Patent Object B	Backward citation	4916
Patent B	Patent A	Forward citation	4918
Patent D	Patent B	Forward citation	4920
Non-Patent Object A	Patent B	Forward citation	4922
Patent C	Patent A	Forward citation	4924
Non-Patent Object B	Patent C	Forward citation	4926
Non-Patent Object C	Patent A	Forward citation	4928
Non-Patent Object D	Patent A	Forward citation	4930

FIG. 49

Relationship Type Table 5002

Rel_Type_ID	Relationship Type Name	Grouping
Rel_Type_001	Backward Citation	
Rel_Type_002	Forward Citation	
Rel_Type_003	Original (Parent)	G1
Rel_Type_004	Continuation	G1
Rel_Type_005	Continuation-In-Part	G1
Rel_Type_006	Divisional	G1
Rel_Type_007	Reissue	G1
Rel_Type_008	Reexamination	G1
Rel_Type_009	Design	G1
Rel_Type_010	Utility	G1
Rel_Type_011	SIR (Statutory Invention Registration)	G1
Rel_Type_012	Defensive Publication	G1
Rel_Type_013	Invention Disclosure Materials	G1
Rel_Type_014	Priority document	G1, G2
Rel_Type_015	PCT	G1, G2
Rel_Type_016	Foreign National Application - Canada	G1, G2
Rel_Type_017	Foreign National Application - Australia	G1, G2
Rel_Type_018	Foreign National Application - United Kingdom	G1, G2
Rel_Type_019	Foreign National Application - Taiwan	G1, G2

FIG. 50A

Rel_Type_020	Foreign National Application - Japan	G1, G2
Rel_Type_021	Foreign National Application - Mexico	G1, G2
Rel_Type_022	Foreign National Application - Brazil	G1, G2
Rel_Type_023	A1	G1, G2
Rel_Type_024	A2	G1, G2
Rel_Type_025	A3	G1, G2
Rel_Type_026	A4	G1, G2
Rel_Type_027	Technically related - Base	G3
Rel_Type_028	Technically related - Improvement	G3
Rel_Type_029	Claim - Dominant	
Rel_Type_030	Claim - Subservient	
Rel_Type_031	Claim - Improvement	
Rel_Type_032	Claim - Independent	
Rel_Type_033	Claim - Dependent	
Rel_Type_034	Trademark	G1
Rel_Type_035	Copyright	G1
Rel_Type_036	Trade Secret	G1

FIG. 50B

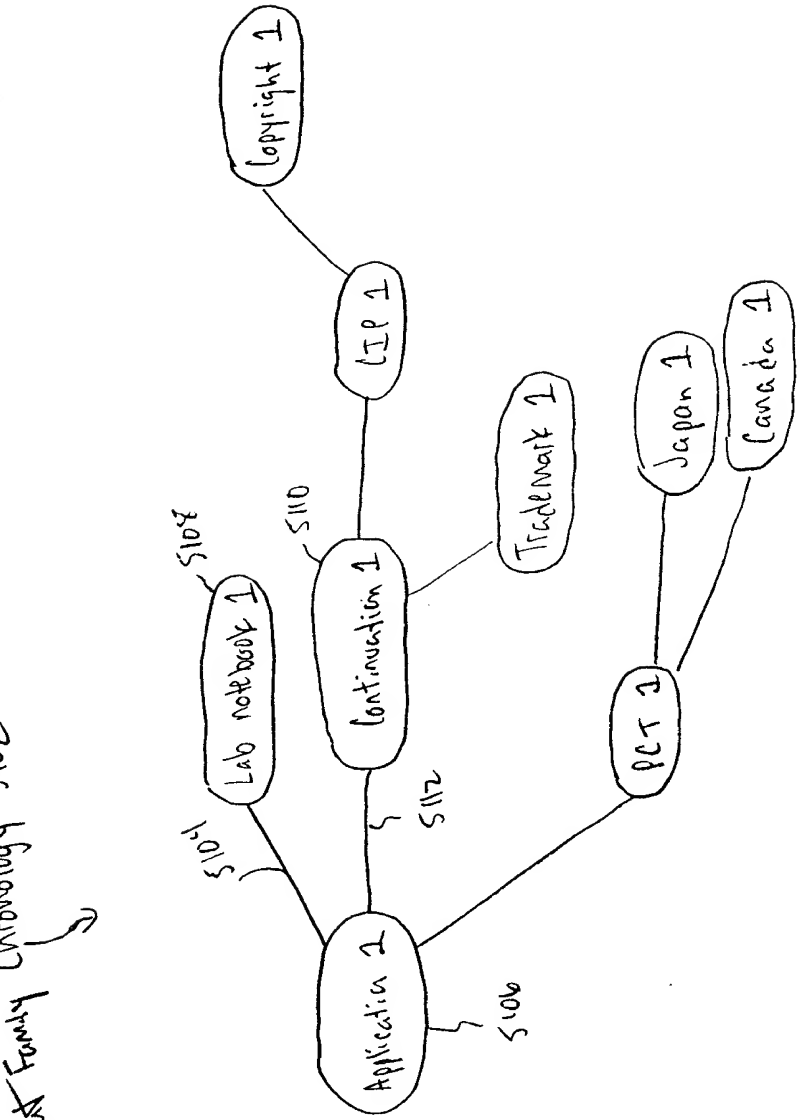

Docket Family Chronology 5102
Application 1
5106
Lab notebook 1
5104
5108-1
Continuation 1
5110
5112
CIP 1
Copyright 1
Trademark 1
PCT 1
Japan 1
Canada 1

FIG. 5I

Data Object Family Table 5202

From	To	Relationship Type	
Application 1	Lab Notebook 1	Invention Disclosure Materials	5204
Application 1	Continuation 1	Continuation	5206
Continuation 1	CIP 1	Continuation-In-Part	5208
CIP 1	Copyright 1	Copyright	5210
Continuation 1	Trademark 1	Trademark	5212
Application 1	PCT 1	PCT	5214
PCT 1	Japan 1	Foreign National Application - Japan	5216
PCT 1	Canada 1	Foreign National Application - Canada	5218

FIG. 52

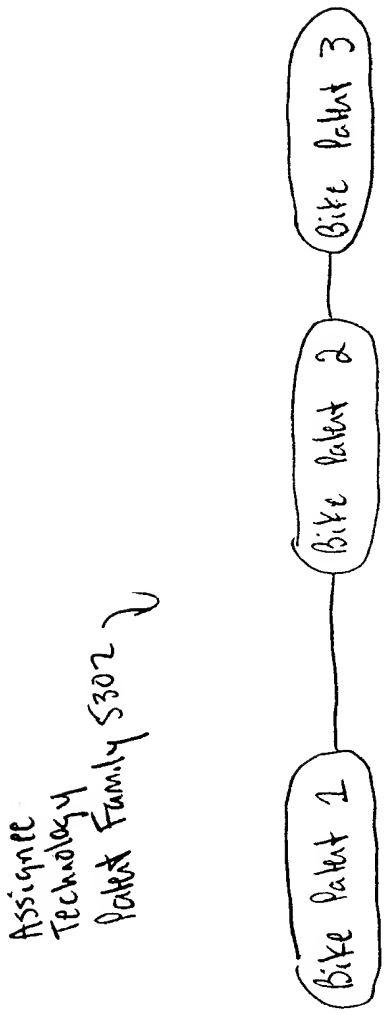

Assignee
Technology
Patent Family 5302
Bike Patent 1
Bike Patent 2
Bike Patent 3

FIG. 53

Data Object Family Table 5402

From	To	Relationship Type	
Bike Patent 1	Bike Patent 2	Technically related - Improvement	5404
Bike Patent 2	Bike Patent 1	Technically related - Base	5406
Bike Patent 2	Bike Patent 3	Technically related - Improvement	5408
Bike Patent 1	Bike Patent 2	Claim - Subservient	5410
Bike Patent 1	Bike Patent 2	Claim - Improvement	5412
Bike Patent 2	Bike Patent 1	Claim - Dominant	5414

FIG. 54

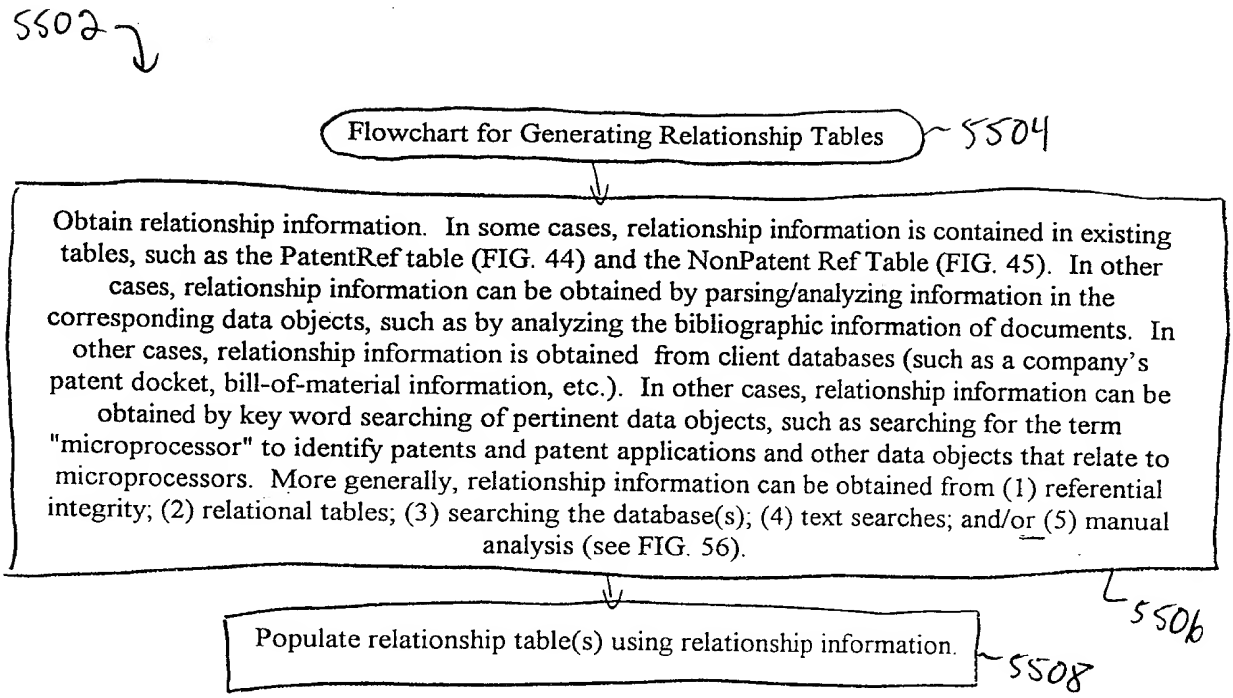

5502
Flowchart for Generating Relationship Tables
5504
Obtain relationship information. In some cases, relationship information is contained in existing tables, such as the PatentRef table (FIG. 44) and the NonPatent Ref Table (FIG. 45). In other cases, relationship information can be obtained by parsing/analyzing information in the corresponding data objects, such as by analyzing the bibliographic information of documents. In other cases, relationship information is obtained from client databases (such as a company's patent docket, bill-of-material information, etc.). In other cases, relationship information can be obtained by key word searching of pertinent data objects, such as searching for the term "microprocessor" to identify patents and patent applications and other data objects that relate to microprocessors. More generally, relationship information can be obtained from (1) referential integrity; (2) relational tables; (3) searching the database(s); (4) text searches; and/or (5) manual analysis (see FIG. 56).
5506
Populate relationship table(s) using relationship information.
5508

FIG. 55

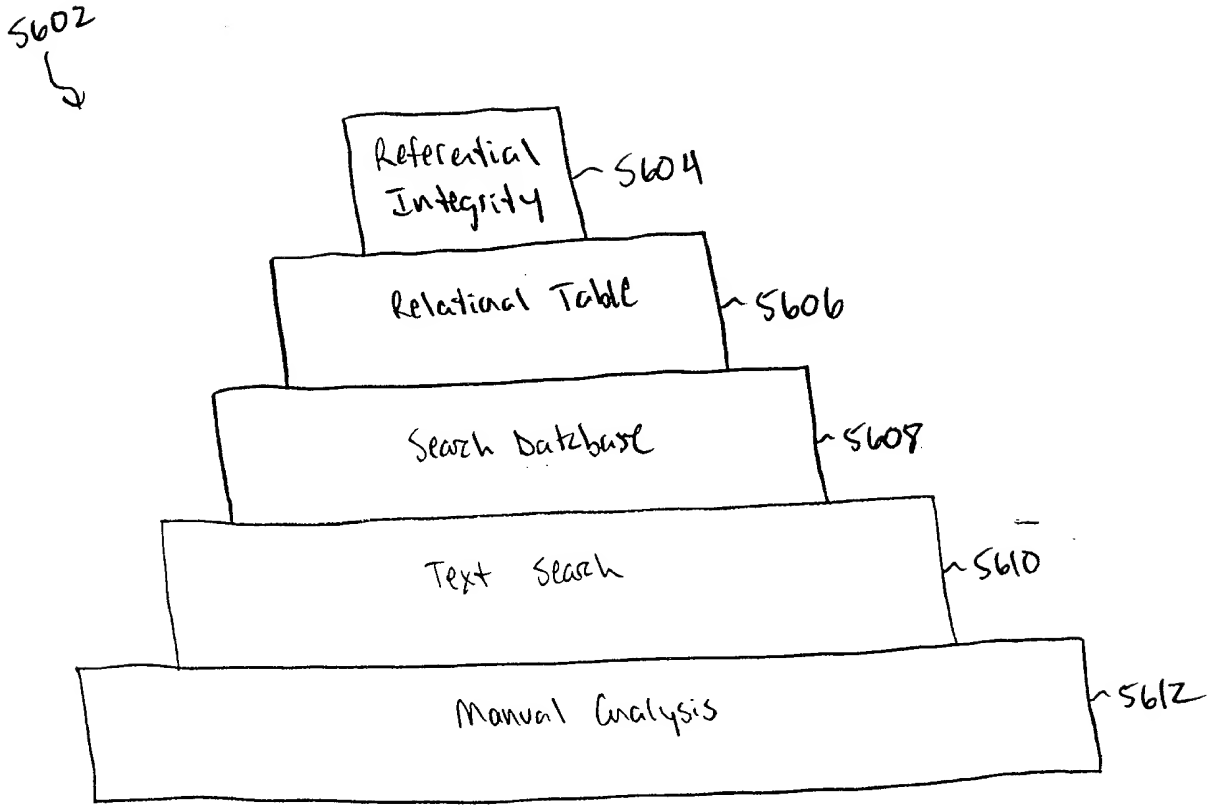

5602
Referential Integrity
5604
Relational Table
5606
Search Database
5608
Text Search
5610
Manual Analysis
5612

FIG. 56

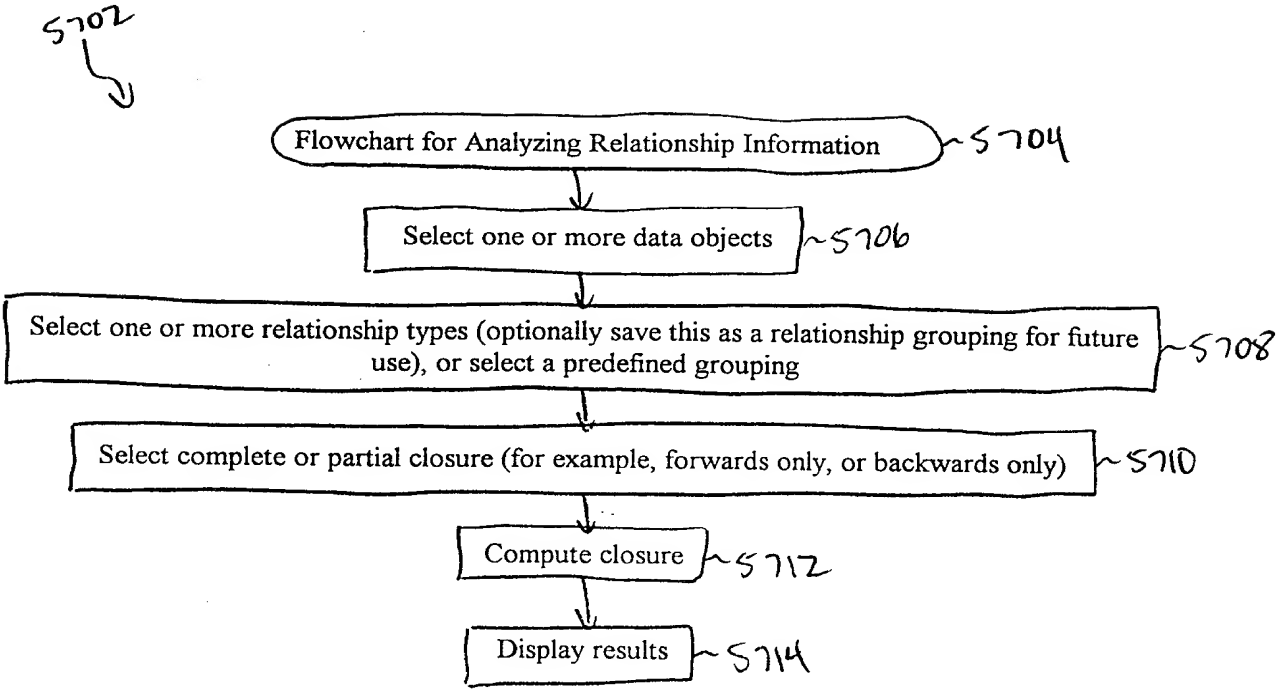

5702
Flowchart for Analyzing Relationship Information
5704
Select one or more data objects
5706
Select one or more relationship types (optionally save this as a relationship grouping for future use), or select a predefined grouping
5708
Select complete or partial closure (for example, forwards only, or backwards only)
5710
Compute closure
5712
Display results
5714

FIG. 57A

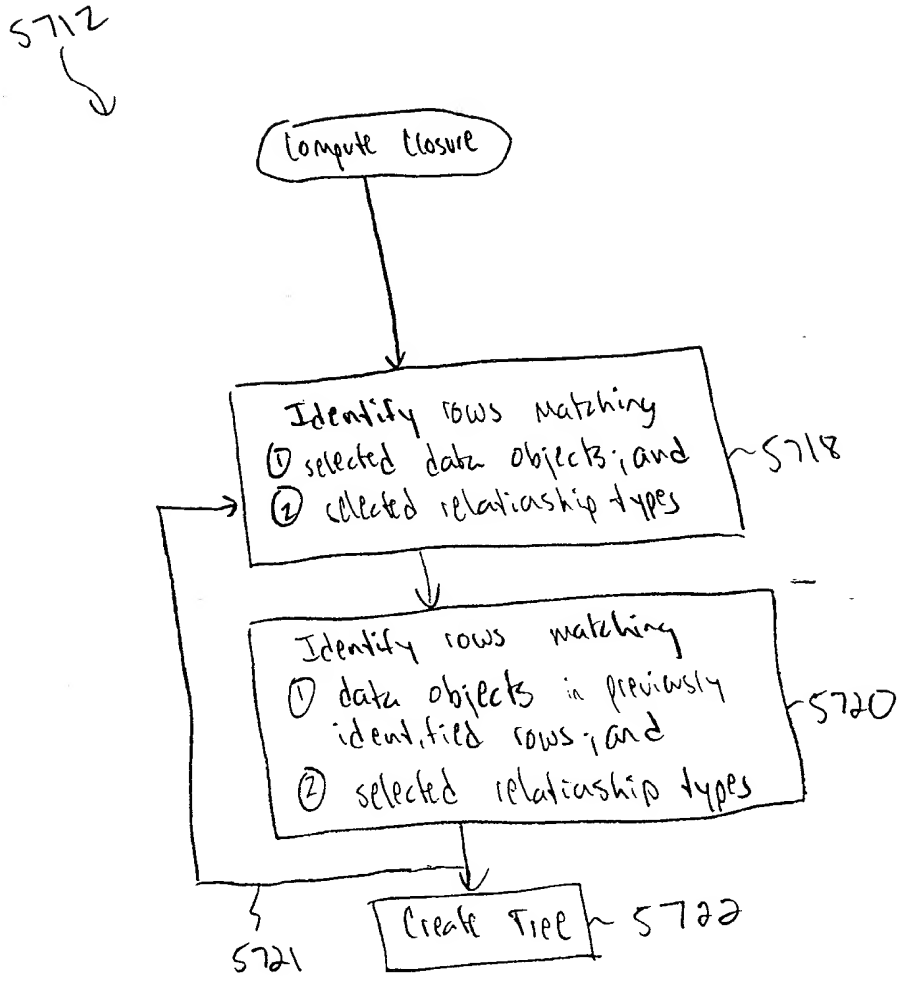

5712
Compute Closure
Identify rows matching
① selected data objects; and
② selected relationship types
5718
Identify rows matching
① data objects in previously identified rows; and
② selected relationship types
5720
5721
Create Tree
5722

FIG. 57B

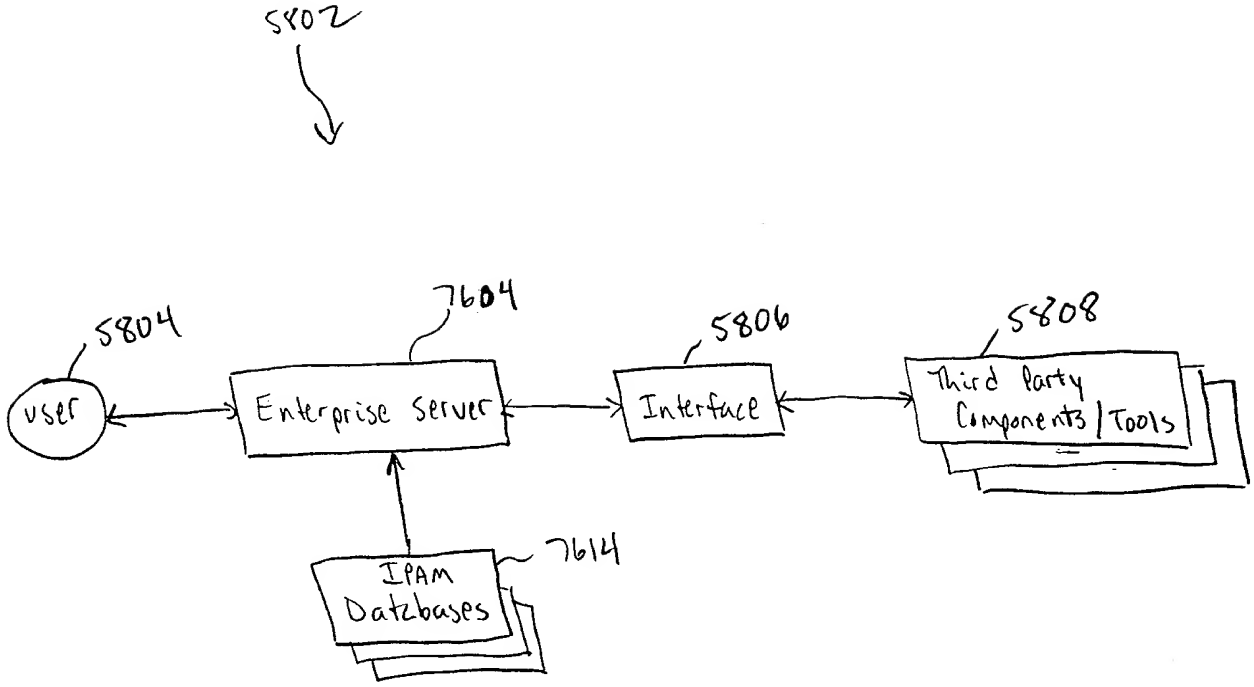

5802
5804
7604
5806
5808
User
Enterprise Server
Interface
Third Party Components / Tools
IPAM Databases
7614

FIG. 58

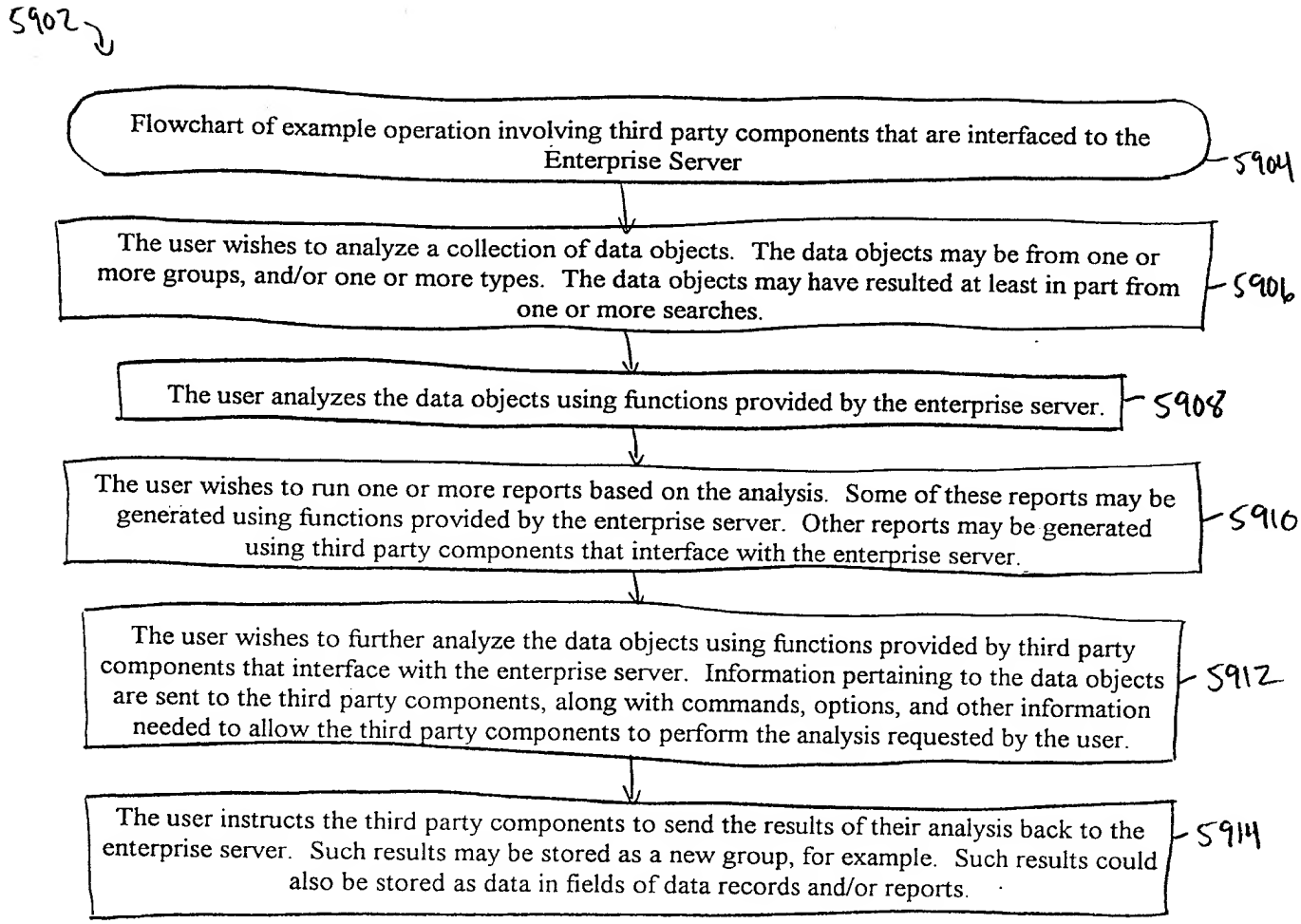

5902
Flowchart of example operation involving third party components that are interfaced to the Enterprise Server
5904
The user wishes to analyze a collection of data objects. The data objects may be from one or more groups, and/or one or more types. The data objects may have resulted at least in part from one or more searches.
5906
The user analyzes the data objects using functions provided by the enterprise server.
5908
The user wishes to run one or more reports based on the analysis. Some of these reports may be generated using functions provided by the enterprise server. Other reports may be generated using third party components that interface with the enterprise server.
5910
The user wishes to further analyze the data objects using functions provided by third party components that interface with the enterprise server. Information pertaining to the data objects are sent to the third party components, along with commands, options, and other information needed to allow the third party components to perform the analysis requested by the user.
5912
The user instructs the third party components to send the results of their analysis back to the enterprise server. Such results may be stored as a new group, for example. Such results could also be stored as data in fields of data records and/or reports.
5914

FIG. 59

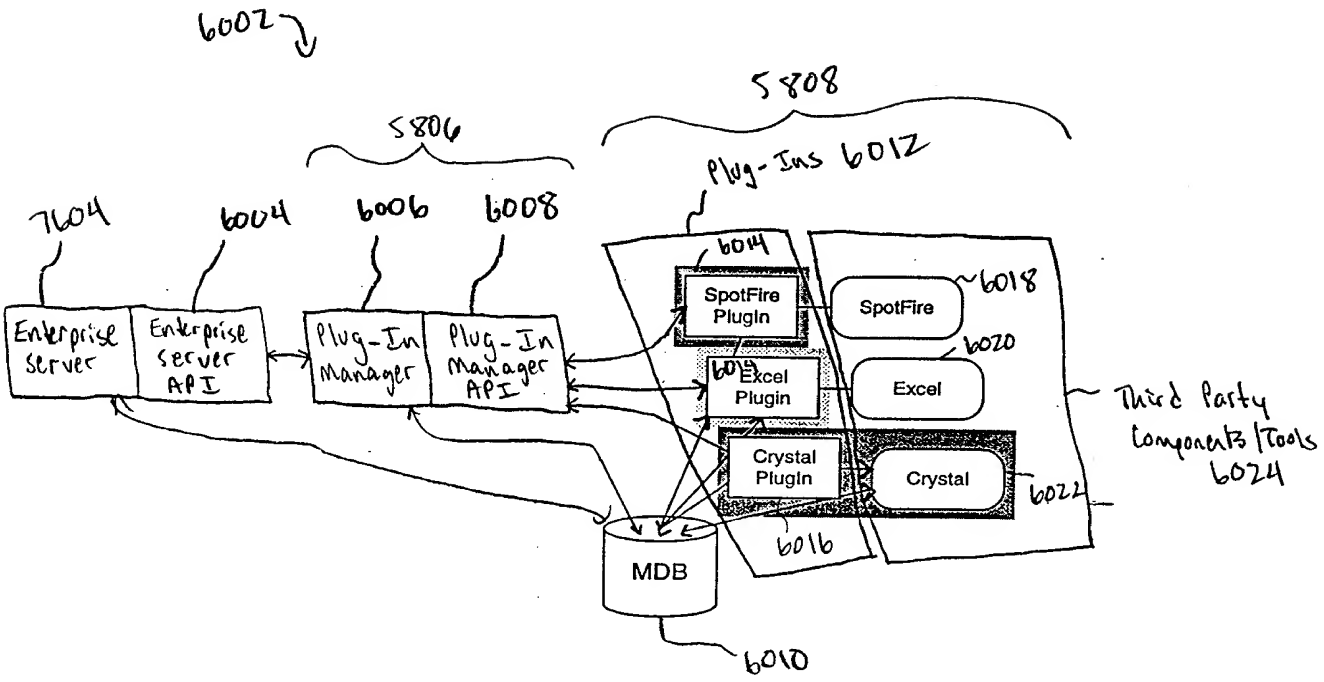

6002
5808
5806
Plug-Ins 6012
7604
6004
6006
6008
6014
SpotFire Plugin
SpotFire
6018
6020
Excel Plugin
Excel
Third Party Components/Tools 6024
Crystal Plugin
Crystal
6022
6016
MDB
6010
Enterprise Server
Enterprise Server API
Plug-In Manager
Plug-In Manager API

FIG. 60

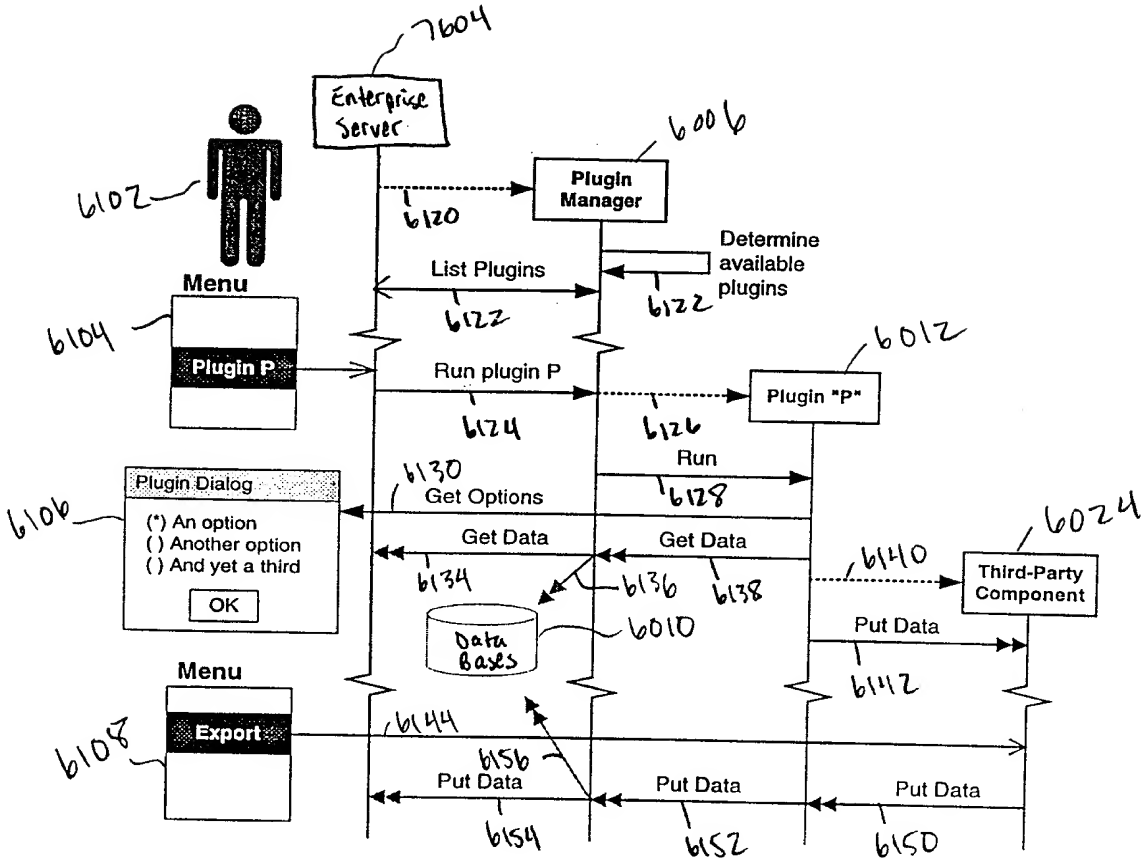

7604
Enterprise Server
6006
Plugin Manager
6102
6120
Determine available plugins
List Plugins
6122
6122
Menu
6104
Plugin P
Run plugin P
6012
Plugin "P"
6124
6126
Run
6128
Plugin Dialog
6106
(*) An option
( ) Another option
( ) And yet a third
OK
6130
Get Options
Get Data
Get Data
6134
6136
6138
6140
6024
Third-Party Component
Data Bases
6010
Put Data
6112
Menu
6108
Export
6144
6156
Put Data
Put Data
Put Data
6154
6152
6150

FIG. 61

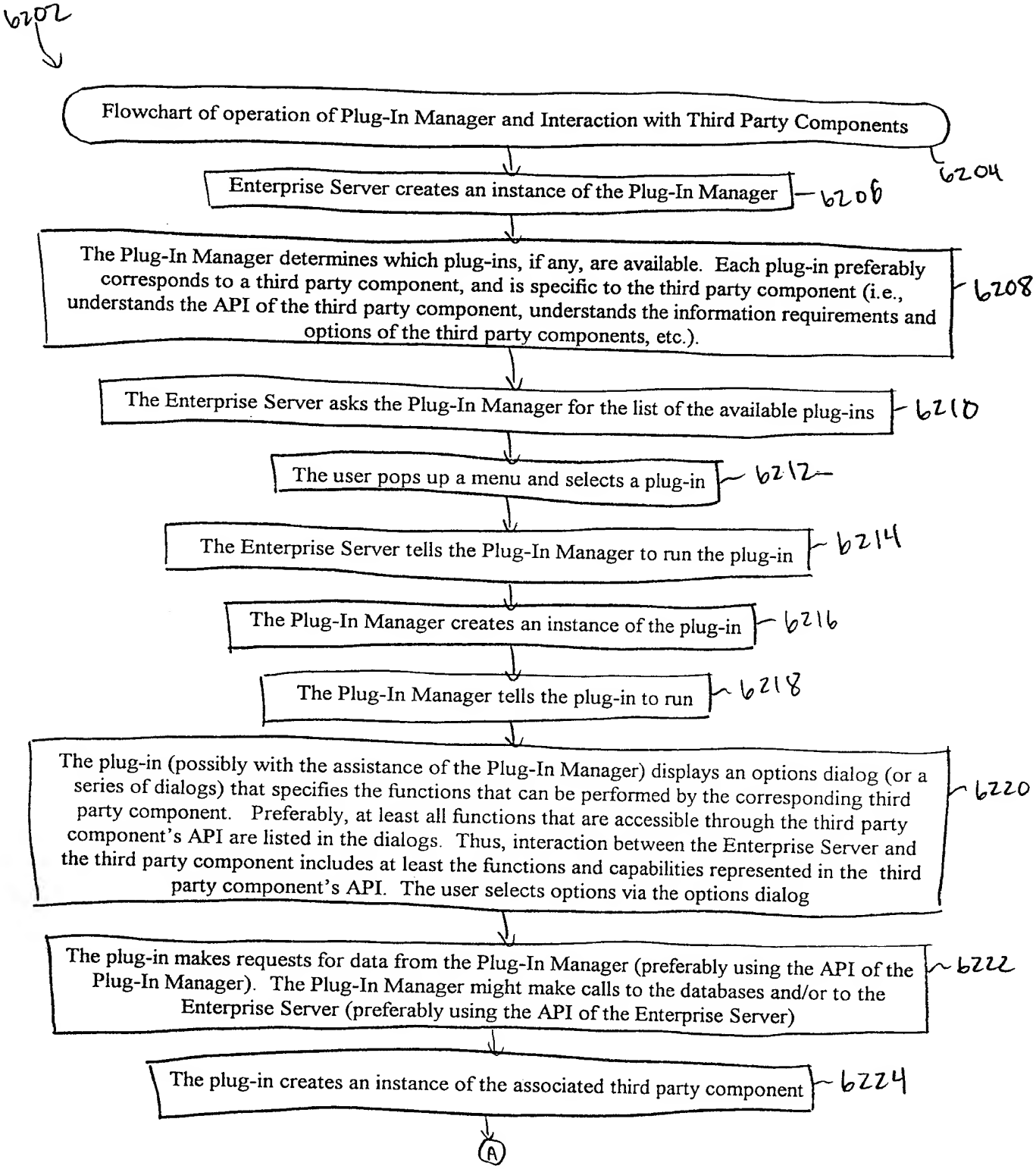

6202
Flowchart of operation of Plug-In Manager and Interaction with Third Party Components
6204
Enterprise Server creates an instance of the Plug-In Manager
6206
The Plug-In Manager determines which plug-ins, if any, are available. Each plug-in preferably corresponds to a third party component, and is specific to the third party component (i.e., understands the API of the third party component, understands the information requirements and options of the third party components, etc.).
6208
The Enterprise Server asks the Plug-In Manager for the list of the available plug-ins
6210
The user pops up a menu and selects a plug-in
6212
The Enterprise Server tells the Plug-In Manager to run the plug-in
6214
The Plug-In Manager creates an instance of the plug-in
6216
The Plug-In Manager tells the plug-in to run
6218
The plug-in (possibly with the assistance of the Plug-In Manager) displays an options dialog (or a series of dialogs) that specifies the functions that can be performed by the corresponding third party component. Preferably, at least all functions that are accessible through the third party component's API are listed in the dialogs. Thus, interaction between the Enterprise Server and the third party component includes at least the functions and capabilities represented in the third party component's API. The user selects options via the options dialog
6220
The plug-in makes requests for data from the Plug-In Manager (preferably using the API of the Plug-In Manager). The Plug-In Manager might make calls to the databases and/or to the Enterprise Server (preferably using the API of the Enterprise Server)
6222
The plug-in creates an instance of the associated third party component
6224
A

FIG. 62A

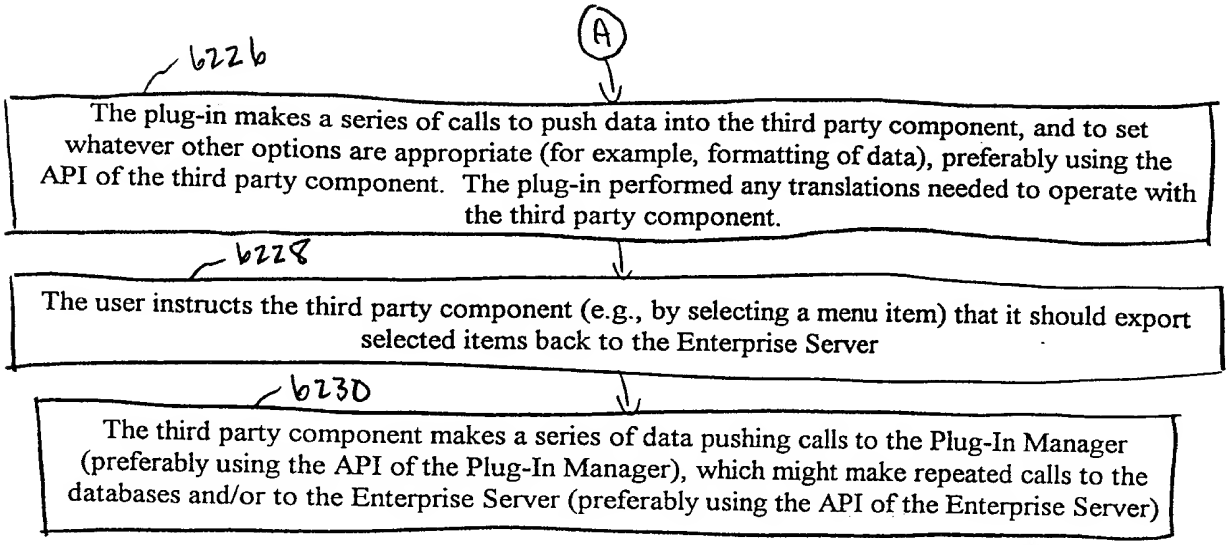

A
6226
The plug-in makes a series of calls to push data into the third party component, and to set whatever other options are appropriate (for example, formatting of data), preferably using the API of the third party component. The plug-in performed any translations needed to operate with the third party component.
6228
The user instructs the third party component (e.g., by selecting a menu item) that it should export selected items back to the Enterprise Server
6230
The third party component makes a series of data pushing calls to the Plug-In Manager (preferably using the API of the Plug-In Manager), which might make repeated calls to the databases and/or to the Enterprise Server (preferably using the API of the Enterprise Server)

FIG. 62B

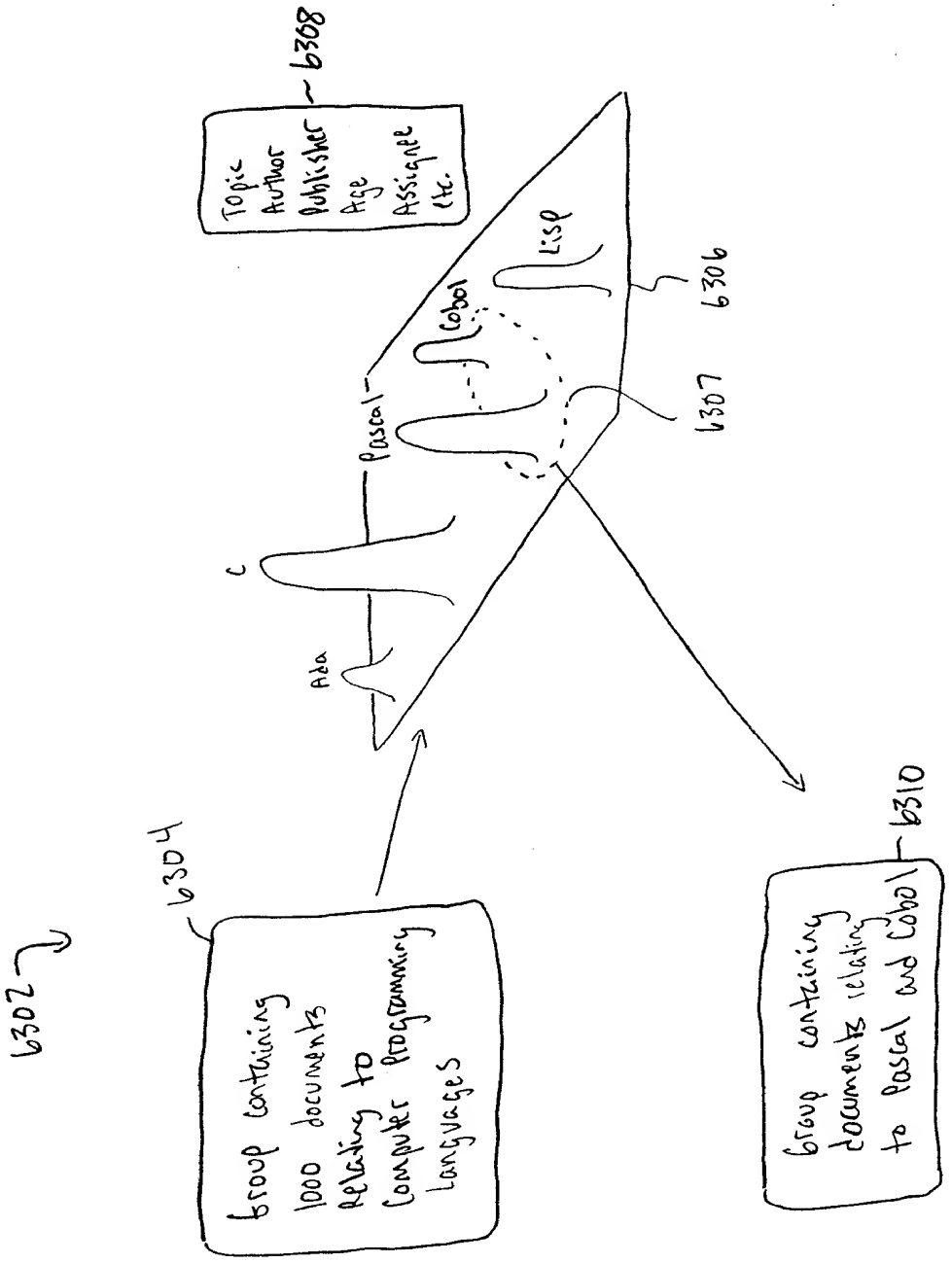

6302
6304
Group containing 1000 documents relating to Computer Programming Languages
Ada
C
Pascal
Cobol
Lisp
6306
6307
6308
Topic
Author
Publisher
Age
Assignee
Etc.
6310
Group containing documents relating to Pascal and Cobol

FIG. 63

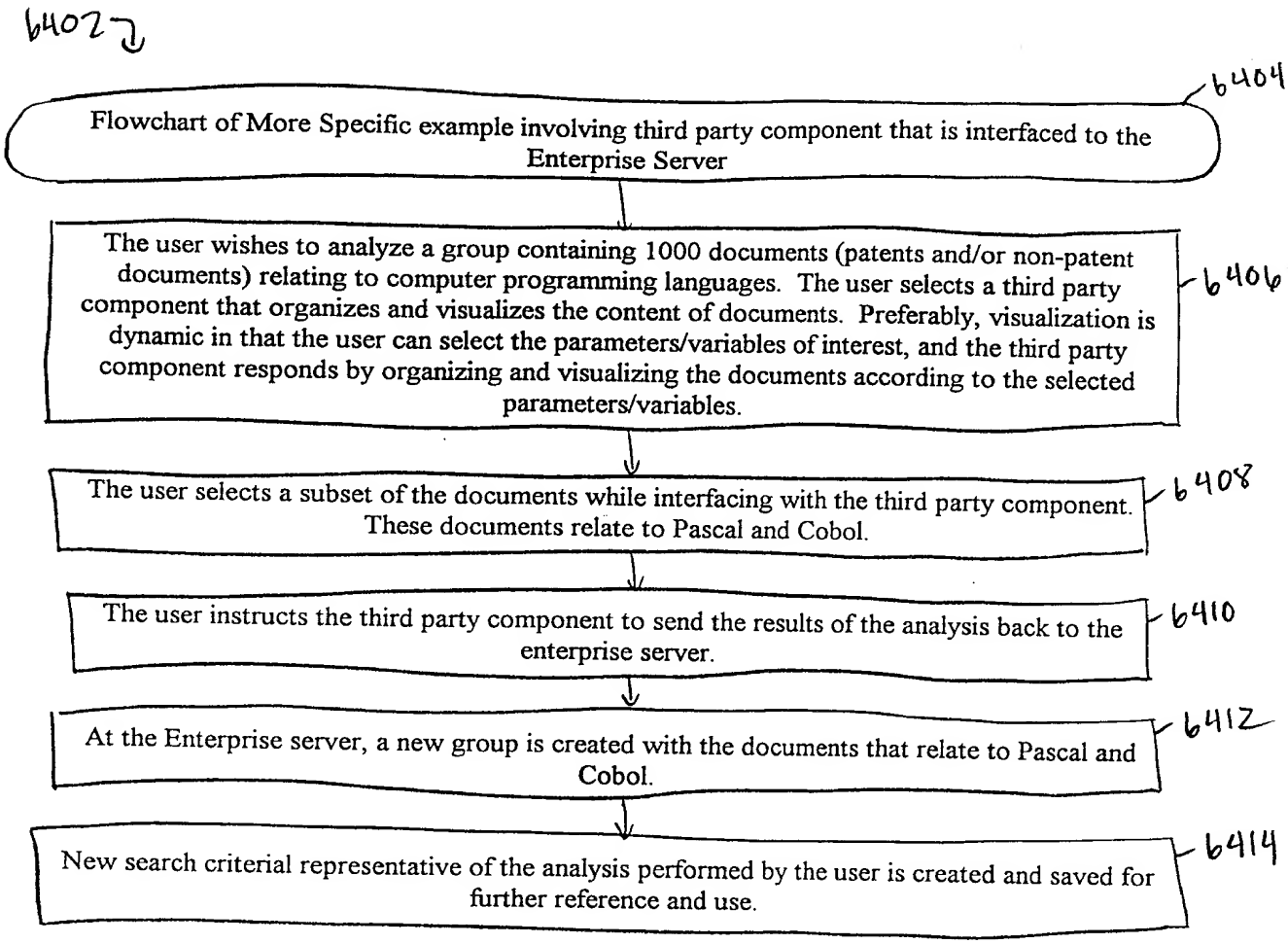

6402
6404
Flowchart of More Specific example involving third party component that is interfaced to the Enterprise Server
The user wishes to analyze a group containing 1000 documents (patents and/or non-patent documents) relating to computer programming languages. The user selects a third party component that organizes and visualizes the content of documents. Preferably, visualization is dynamic in that the user can select the parameters/variables of interest, and the third party component responds by organizing and visualizing the documents according to the selected parameters/variables.
6406
The user selects a subset of the documents while interfacing with the third party component. These documents relate to Pascal and Cobol.
6408
The user instructs the third party component to send the results of the analysis back to the enterprise server.
6410
At the Enterprise server, a new group is created with the documents that relate to Pascal and Cobol.
6412
New search criterial representative of the analysis performed by the user is created and saved for further reference and use.
6414

FIG. 64

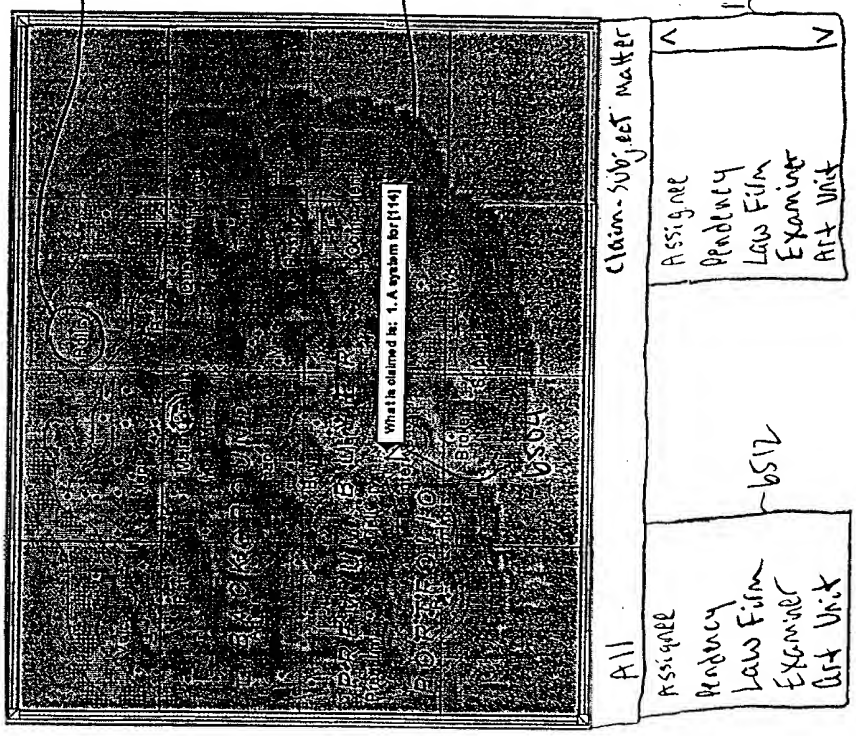

6508
6506
6510
What is claimed is: 1. A system for [[its]]
Claim-Subject matter
All
Assignee
Pendency
Law Firm
Examiner
Art Unit
6512
6502

FIG. 65

6604

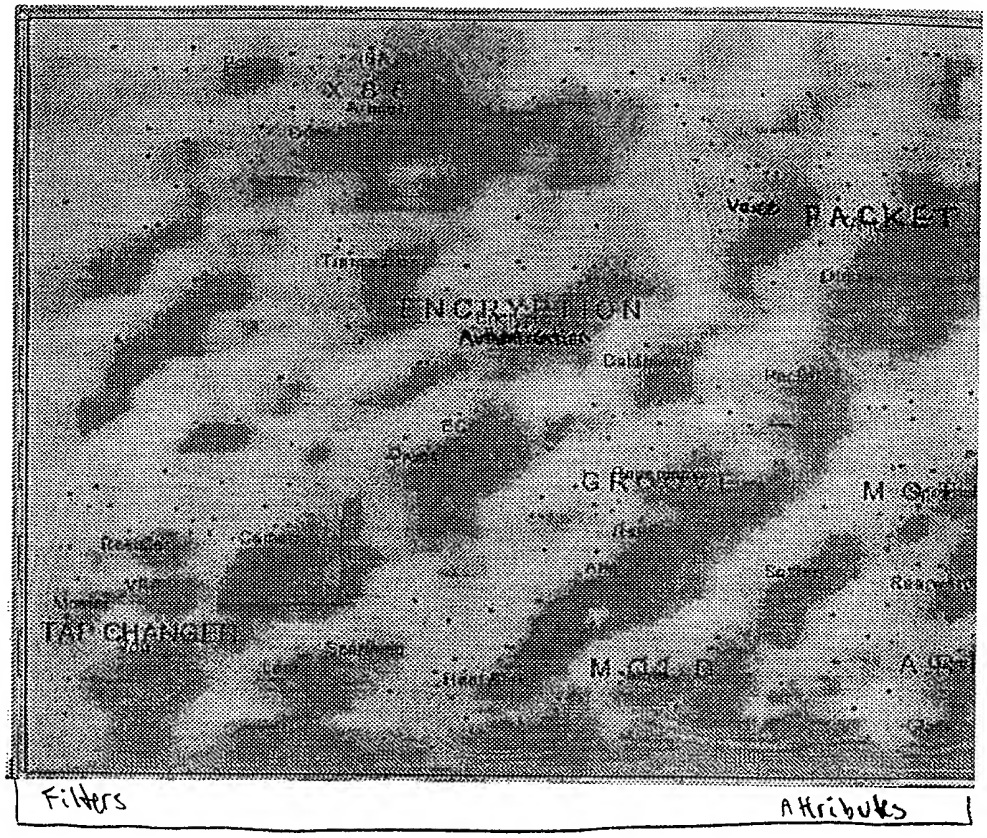


FIG. 66

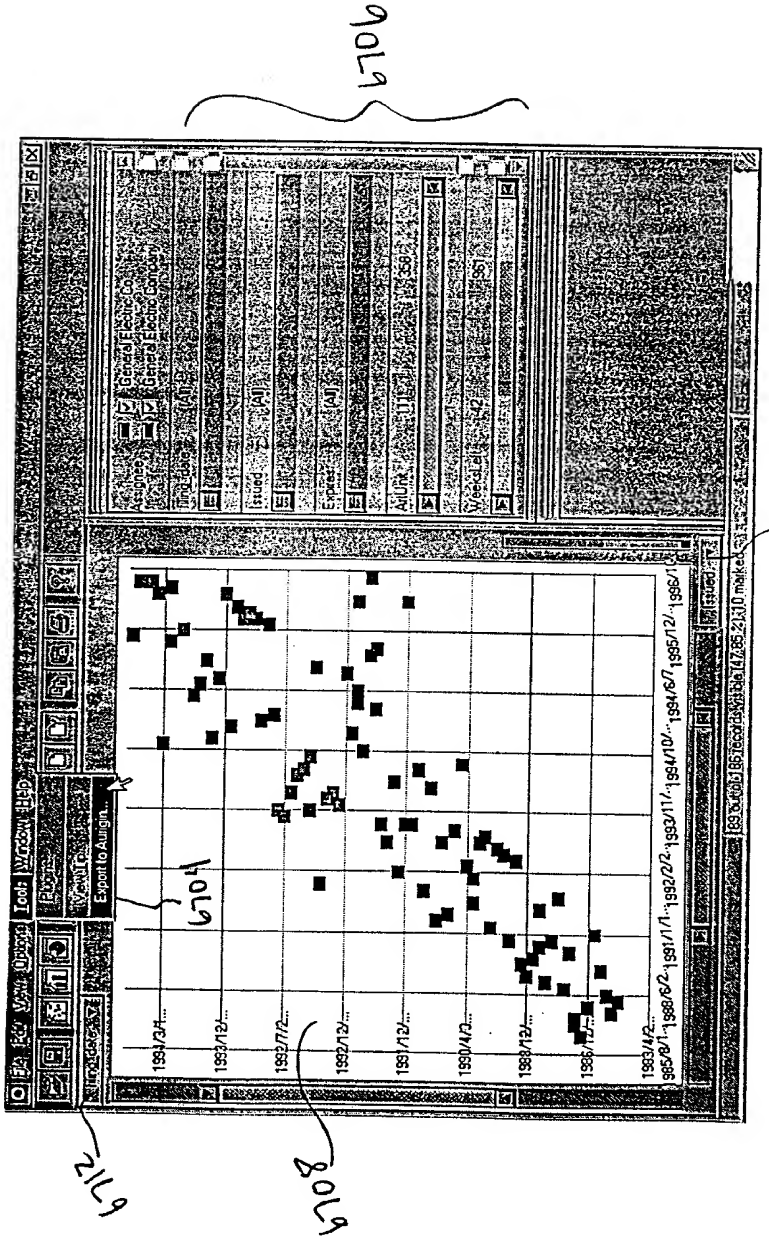

6702
6704
6706
6708
6710
6712
Export to Autogin...
Tools
Window
Help
Assignee
Inventor
Abstract
Claims
Description
Export
Word
Excel
Text
Autogin
WebSelect
Expert
General Electric Co
General Electric Company
1993/3/1...
1993/1/1...
1992/7/2...
1992/1/1...
1991/1/1...
1990/4/2...
1989/1/1...
1988/7/2...
1987/4/2...
1995/8/1-1988/6/2-1991/1/1-1992/2/2-1993/11/-1994/10/...1994/6/7-1995/12/...1996/1...

FIG. 67

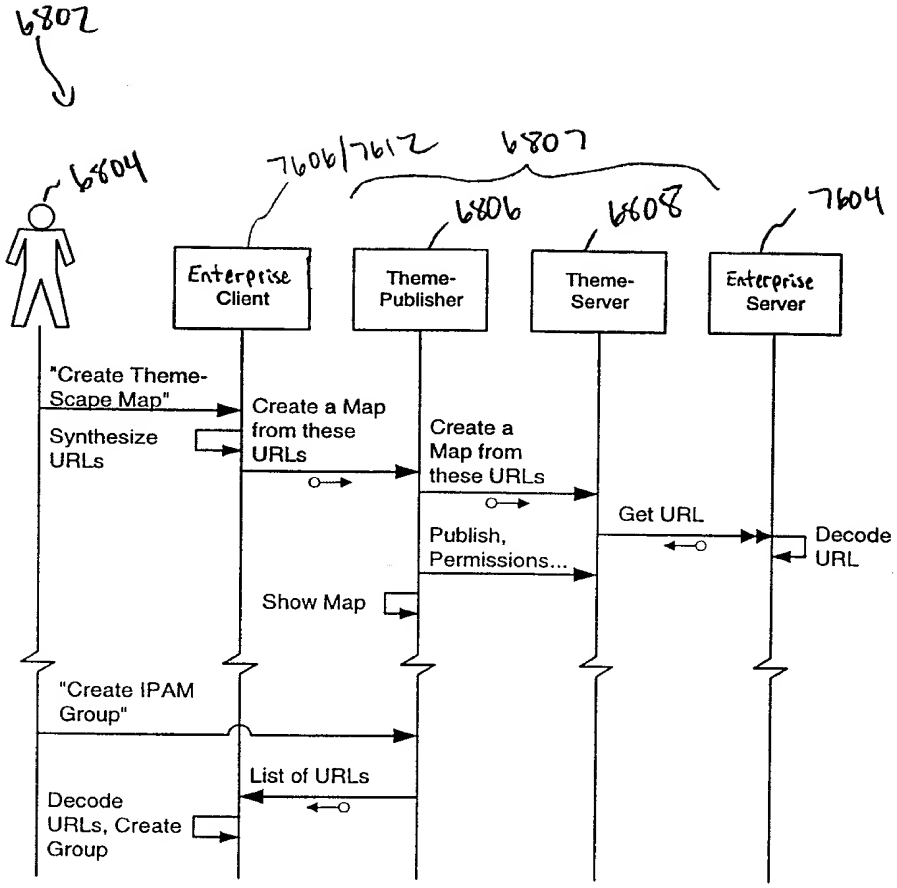

6807
6804
7606/7612
6807
6806
6808
7604
Enterprise Client
Theme-Publisher
Theme-Server
Enterprise Server
"Create Theme-Scape Map"
Synthesize URLs
Create a Map from these URLs
Create a Map from these URLs
Get URL
Decode URL
Publish, Permissions...
Show Map
"Create IPAM Group"
List of URLs
Decode URLs, Create Group

FIG. 68

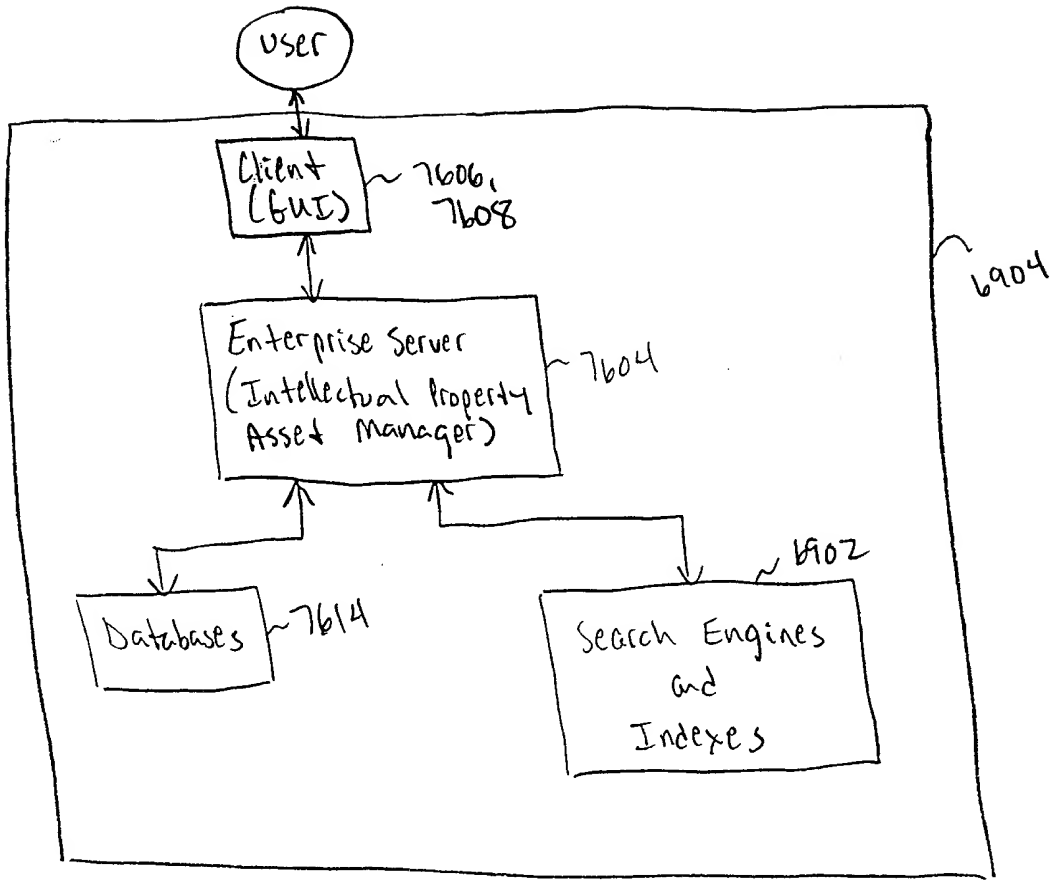

user
Client (GUI)
7606, 7608
6904
Enterprise Server (Intellectual Property Asset Manager)
7604
6902
Databases
7614
Search Engines and Indexes

FIG. 69

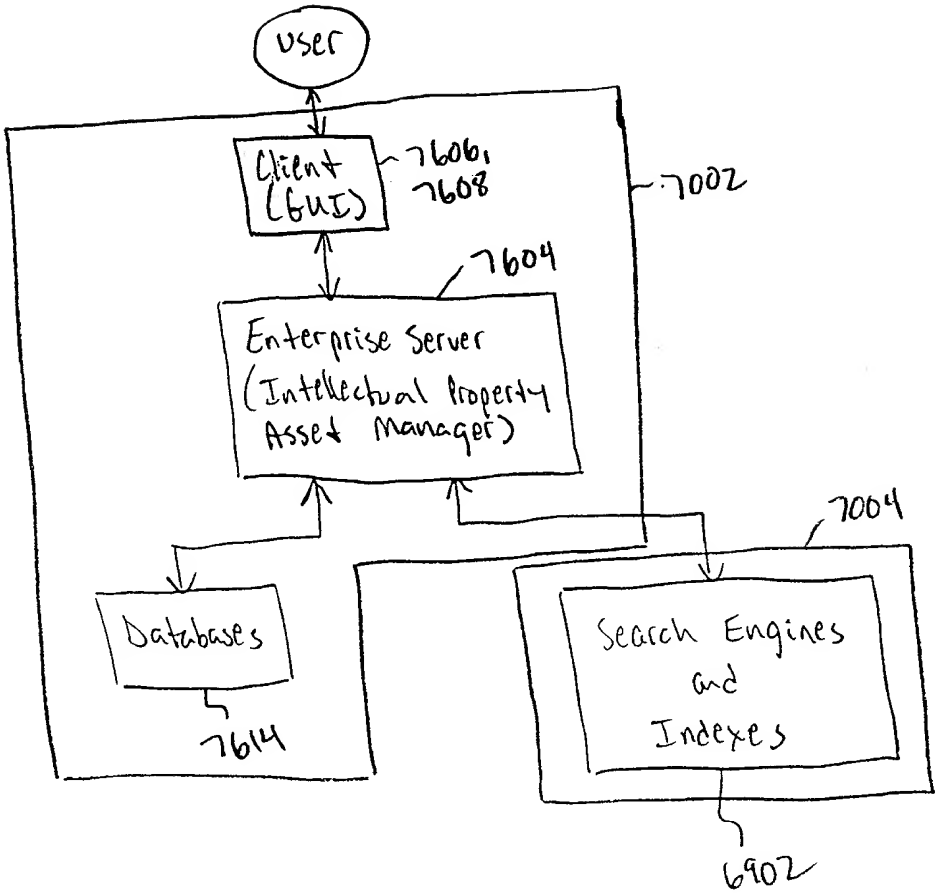

User
Client (GUI)
7606, 7608
7002
7604
Enterprise Server (Intellectual Property Asset Manager)
7004
Databases
7614
Search Engines and Indexes
6902

FIG. 70

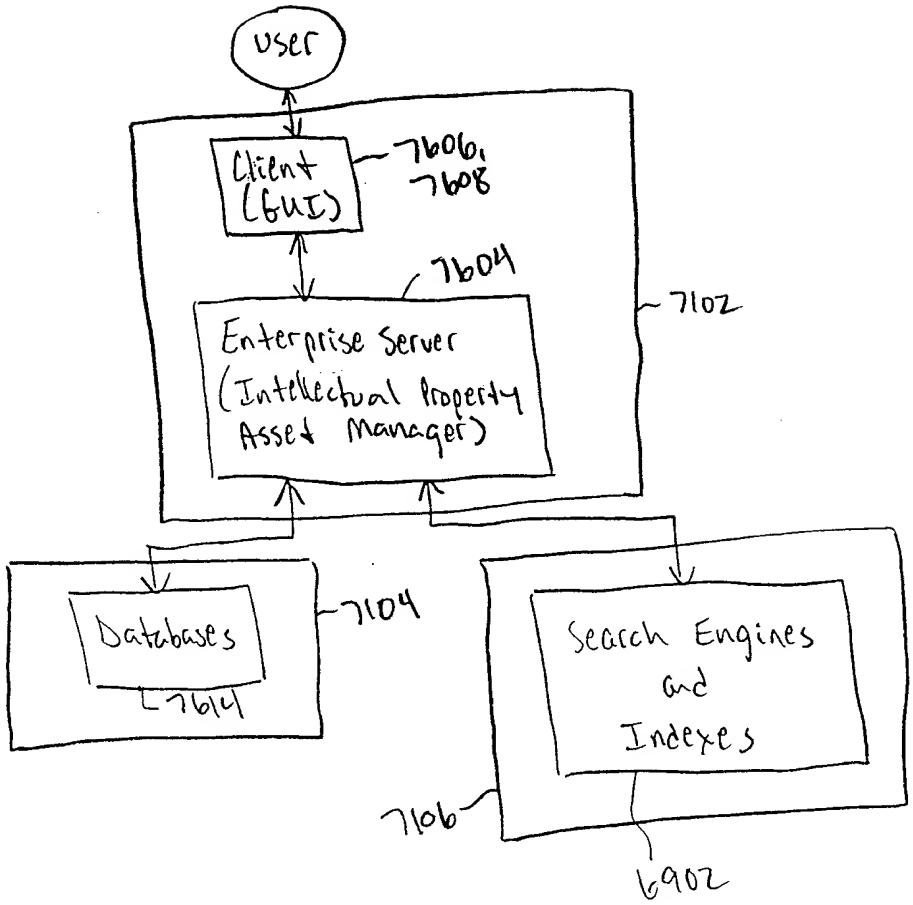

User
Client (GUI)
7606, 7608
7604
Enterprise Server (Intellectual Property Asset Manager)
7102
Databases
7104
7614
Search Engines and Indexes
7106
6902

FIG. 71

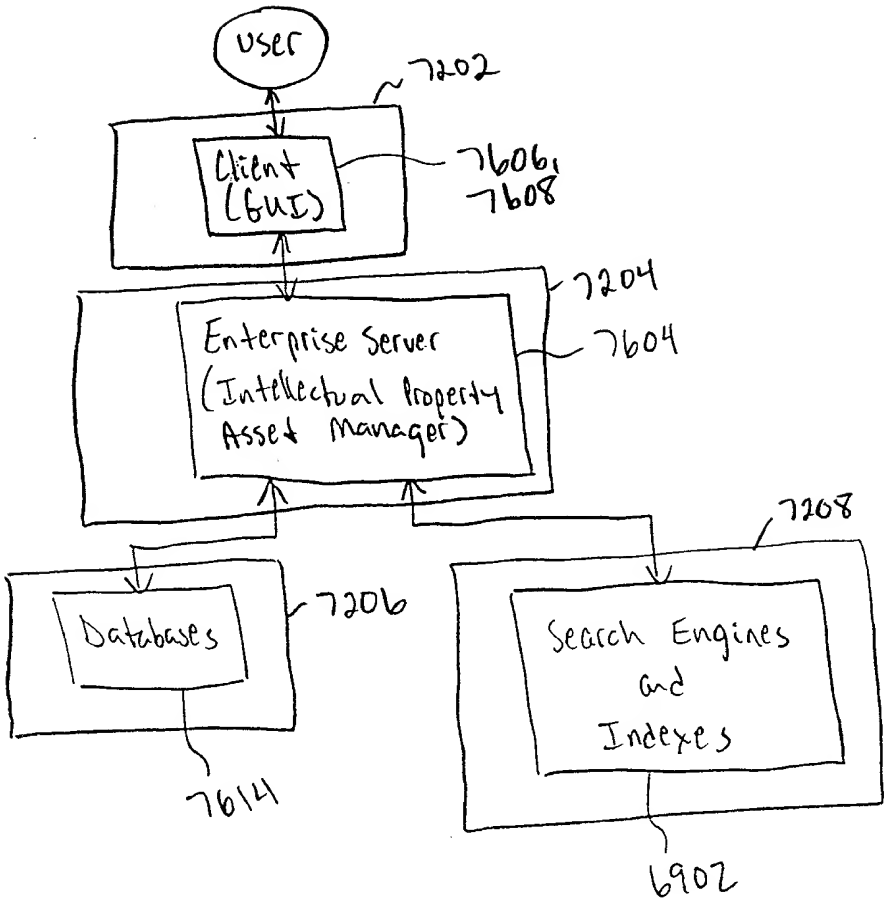

User
7202
Client (GUI)
7606, 7608
7204
Enterprise Server (Intellectual Property Asset Manager)
7604
7208
Databases
7206
Search Engines and Indexes
7614
6902

FIG. 72

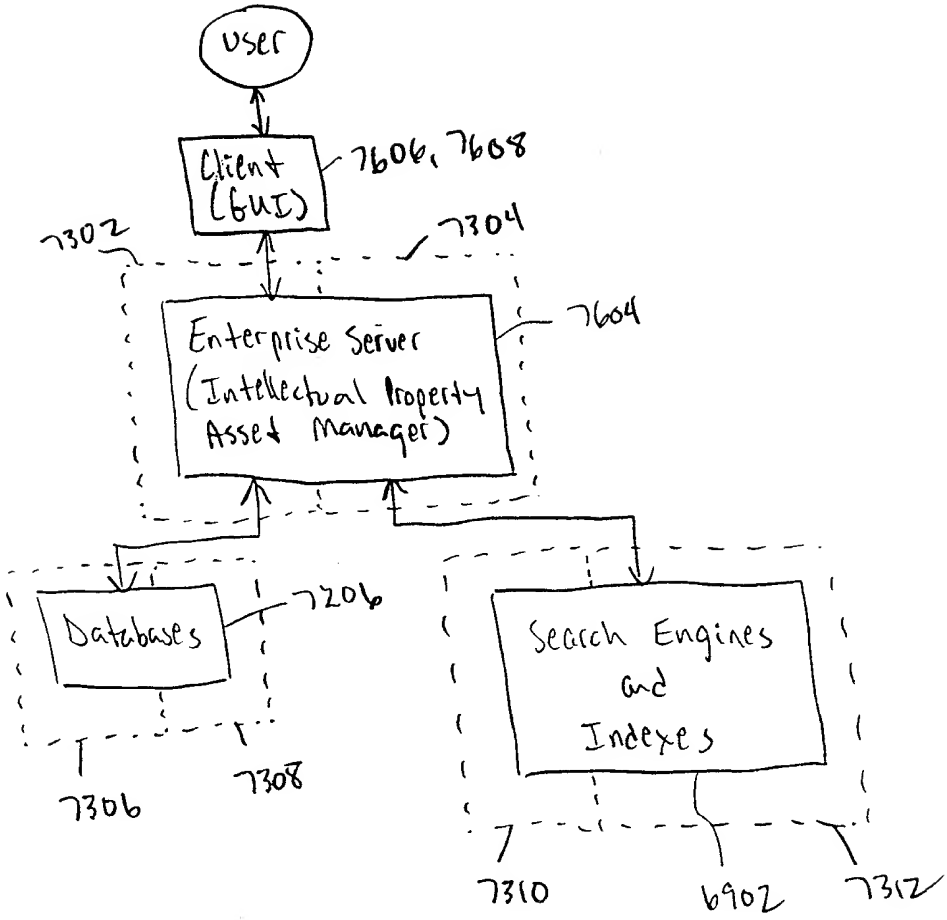

User
Client (GUI)
7606, 7608
7302
7304
Enterprise Server (Intellectual Property Asset Manager)
7604
Databases
7206
Search Engines and Indexes
7306
7308
7310
6902
7312

FIG. 73

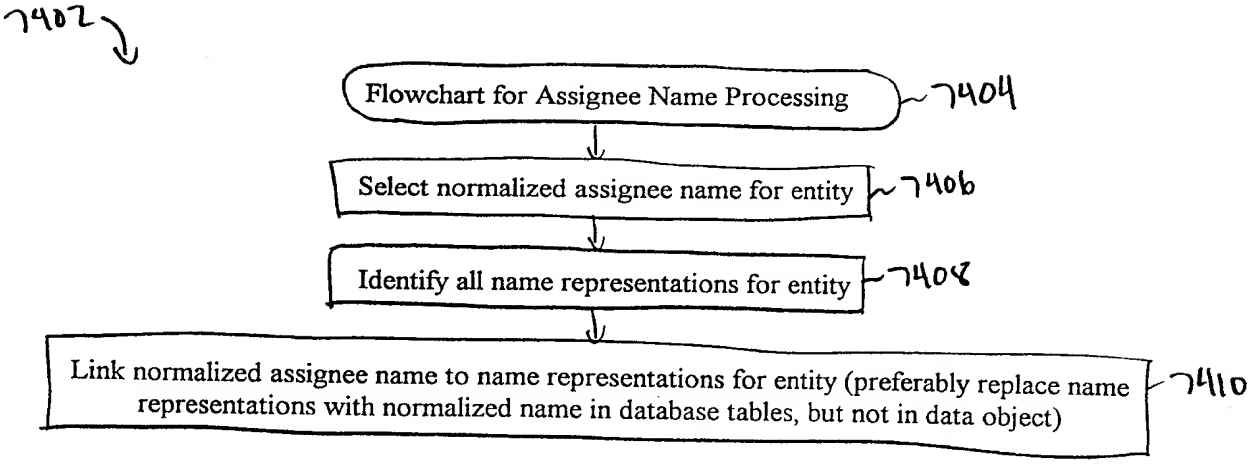

7402
Flowchart for Assignee Name Processing
7404
Select normalized assignee name for entity
7406
Identify all name representations for entity
7408
Link normalized assignee name to name representations for entity (preferably replace name representations with normalized name in database tables, but not in data object)
7410

FIG. 74

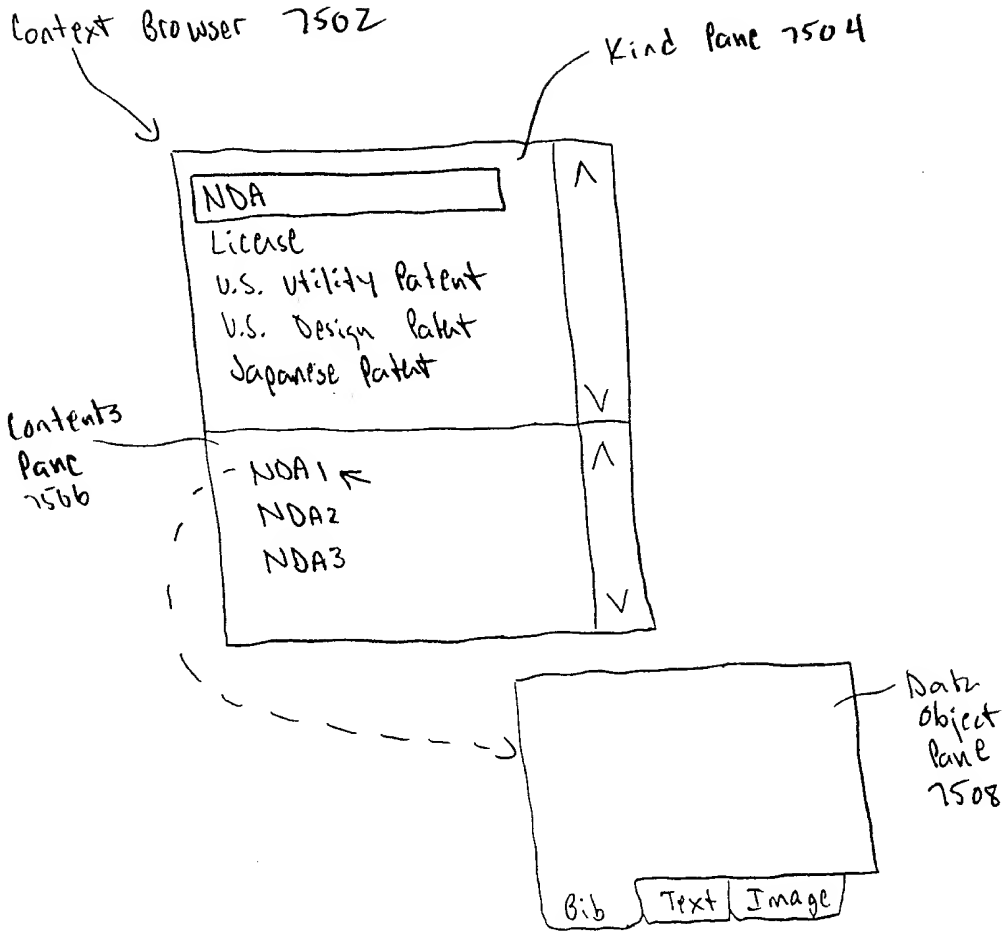

Context Browser 7502
Kind Pane 7504
NDA
License
U.S. Utility Patent
U.S. Design Patent
Japanese Patent
Contents Pane 7506
NDA1
NDA2
NDA3
Data object pane 7508
Bib
Text
Image

FIG. 75

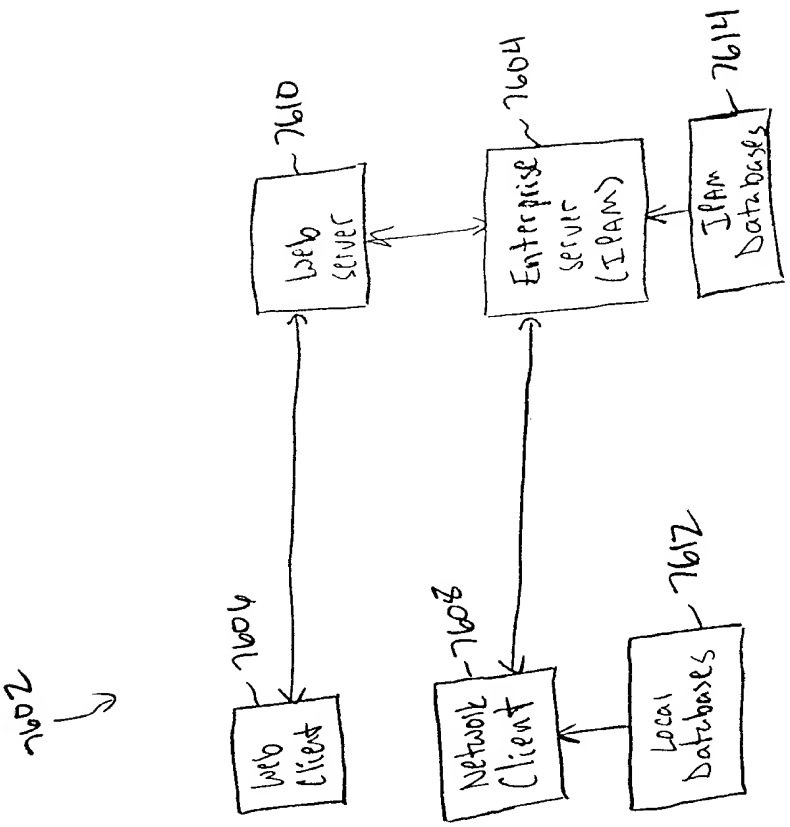

7602
Web Client
7606
Network Client
7608
Local Databases
7612
Web Server
7610
Enterprise Server (IPAM)
7604
IPAM Databases
7614

FIG. 76

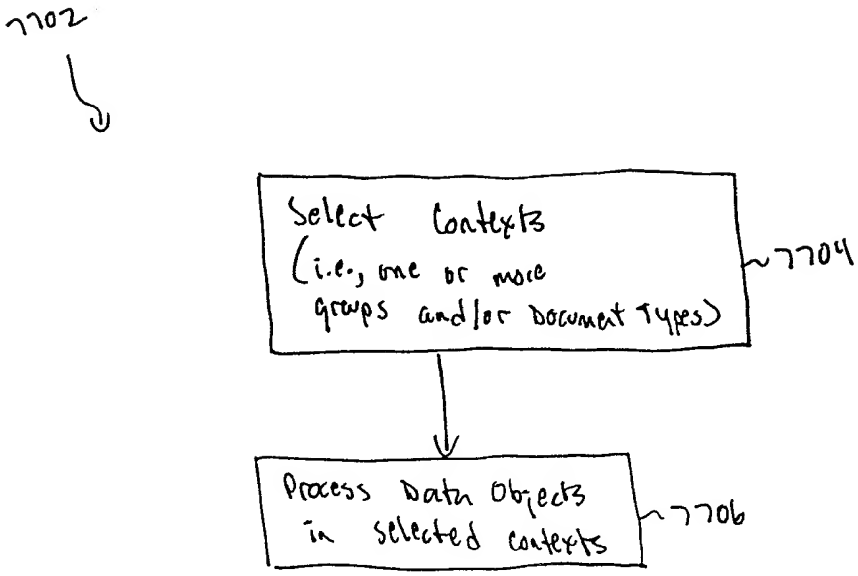

7702
Select Contexts (i.e., one or more groups and/or Document Types)
7704
Process Data Objects in selected contexts
7706

FIG. 77

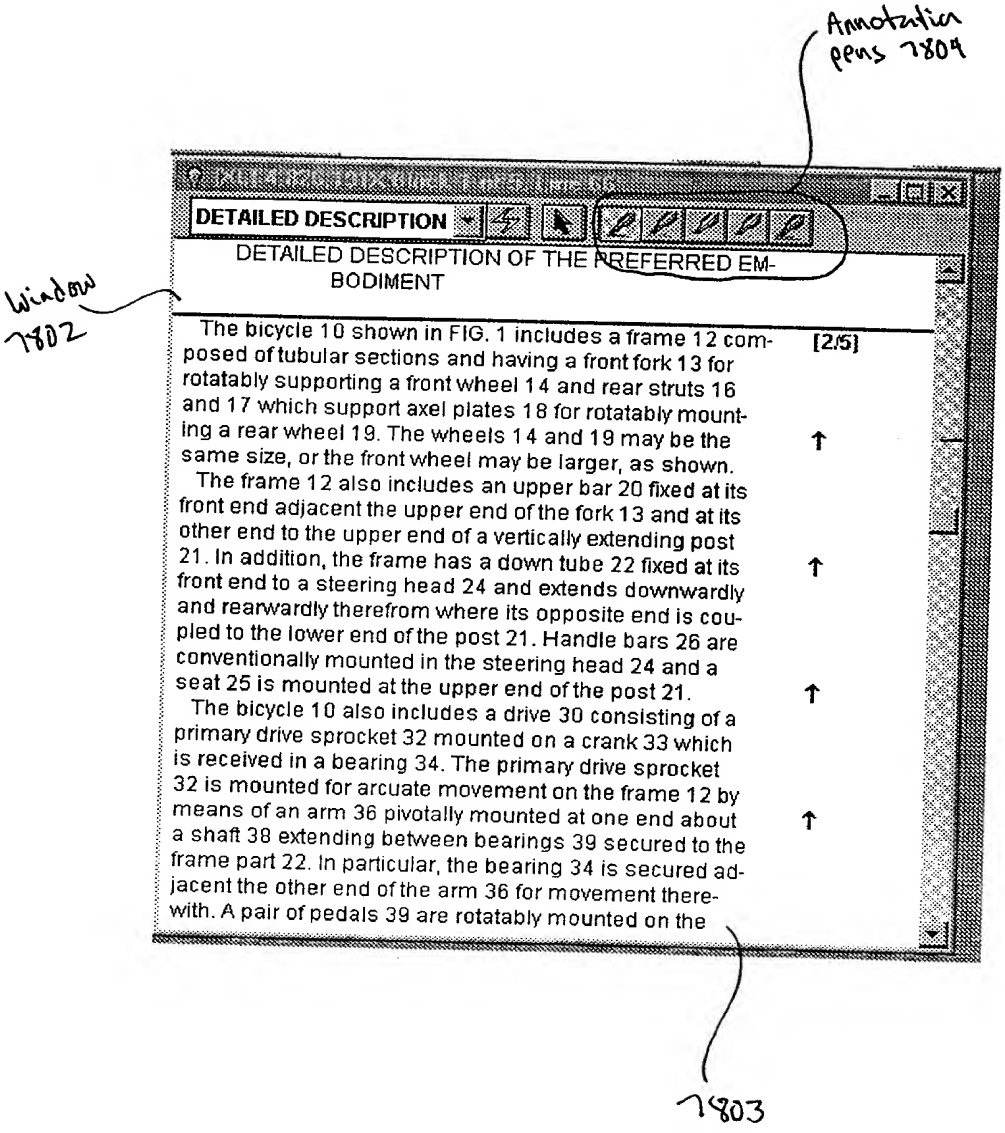

Annotation pens 7804
DETAILED DESCRIPTION
Window 7802
DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT
[2/5]
The bicycle 10 shown in FIG. 1 includes a frame 12 composed of tubular sections and having a front fork 13 for rotatably supporting a front wheel 14 and rear struts 16 and 17 which support axel plates 18 for rotatably mounting a rear wheel 19. The wheels 14 and 19 may be the same size, or the front wheel may be larger, as shown.
The frame 12 also includes an upper bar 20 fixed at its front end adjacent the upper end of the fork 13 and at its other end to the upper end of a vertically extending post 21. In addition, the frame has a down tube 22 fixed at its front end to a steering head 24 and extends downwardly and rearwardly therefrom where its opposite end is coupled to the lower end of the post 21. Handle bars 26 are conventionally mounted in the steering head 24 and a seat 25 is mounted at the upper end of the post 21.
The bicycle 10 also includes a drive 30 consisting of a primary drive sprocket 32 mounted on a crank 33 which is received in a bearing 34. The primary drive sprocket 32 is mounted for arcuate movement on the frame 12 by means of an arm 36 pivotally mounted at one end about a shaft 38 extending between bearings 39 secured to the frame part 22. In particular, the bearing 34 is secured adjacent the other end of the arm 36 for movement therewith. A pair of pedals 39 are rotatably mounted on the
7803

FIG. 78A

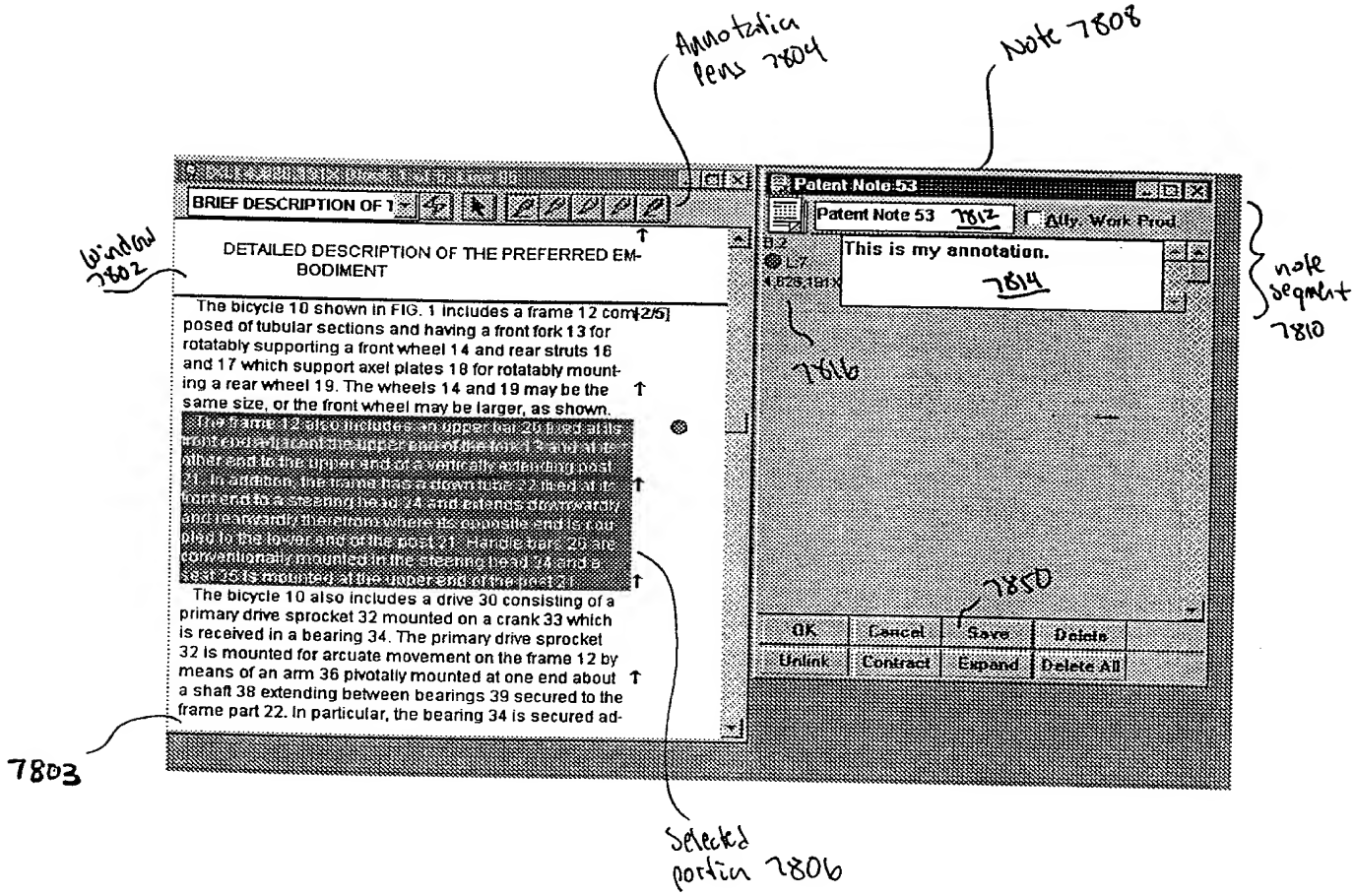

Annotation pens 7804
Note 7808
Window 7802
BRIEF DESCRIPTION OF T
DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT
The bicycle 10 shown in FIG. 1 includes a frame 12 com[2/5]
posed of tubular sections and having a front fork 13 for
rotatably supporting a front wheel 14 and rear struts 16
and 17 which support axel plates 18 for rotatably mounting a rear wheel 19. The wheels 14 and 19 may be the
same size, or the front wheel may be larger, as shown.
The bicycle 10 also includes a drive 30 consisting of a
primary drive sprocket 32 mounted on a crank 33 which
is received in a bearing 34. The primary drive sprocket
32 is mounted for arcuate movement on the frame 12 by
means of an arm 36 pivotally mounted at one end about
a shaft 38 extending between bearings 39 secured to the
frame part 22. In particular, the bearing 34 is secured ad-
Patent Note 53
Patent Note 53 7812
Ally Work Prod
This is my annotation.
7814
7816
note segment 7810
7850
OK
Cancel
Save
Delete
Unlink
Contract
Expand
Delete All
7803
Selected portion 7806

FIG. 78

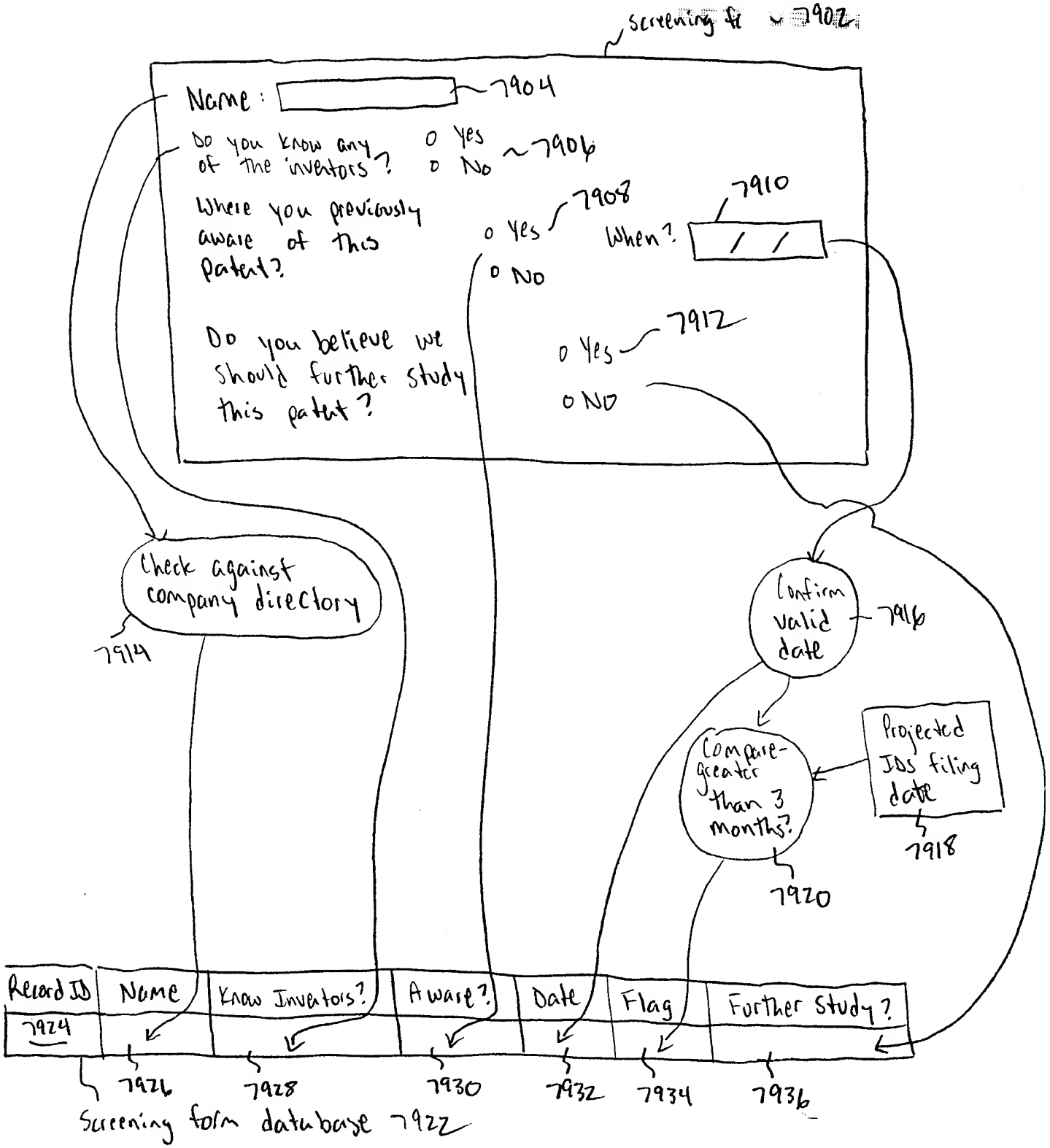

Screening form 7902
Name: 7904
Do you know any of the inventors? o Yes o No 7906
Where you previously aware of this patent? o Yes 7908 o No
When? 7910
Do you believe we should further study this patent? o Yes 7912 o No
Check against company directory 7914
Confirm valid date 7916
Compare-greater than 3 months? 7920
Projected IDS filing date 7918
Record ID 7924
Name 7926
Know Inventors? 7928
Aware? 7930
Date 7932
Flag 7934
Further Study? 7936
Screening form database 7922

FIG. 79

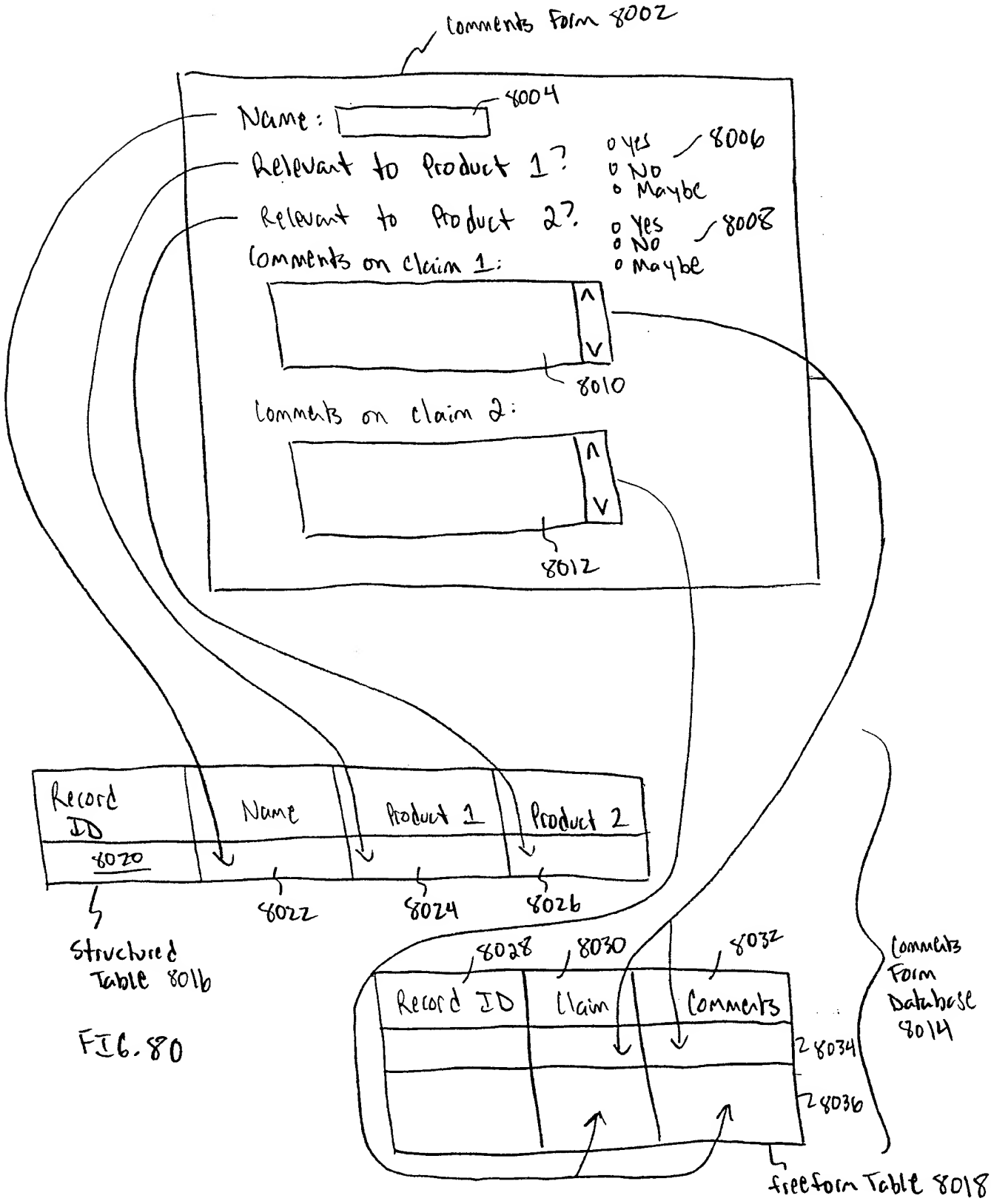

Comments Form 8002
Name:
8004
Relevant to Product 1?
o Yes
o No
o Maybe
8006
Relevant to Product 2?
o Yes
o No
o Maybe
8008
Comments on claim 1:
8010
Comments on claim 2:
8012
Record ID
Name
Product 1
Product 2
8020
8022
8024
8026
Structured Table 8016
8028
8030
8032
Record ID
Claim
Comments
8034
8036
Comments Form Database 8014
Freeform Table 8018

FIG. 80

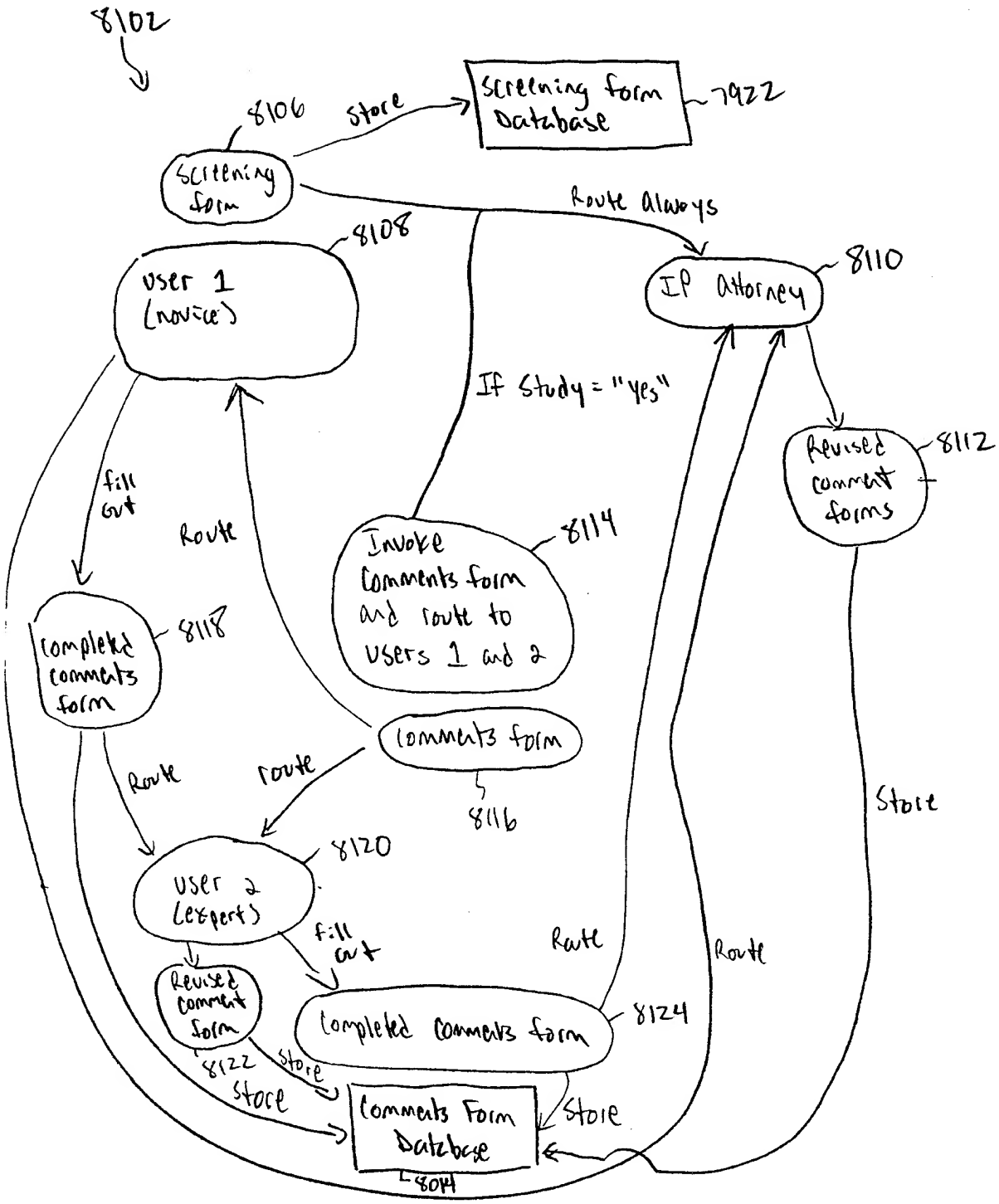

8102
8106
store
Screening form Database
7922
Screening form
Route always
8108
User 1 (novice)
8110
IP Attorney
If Study = "yes"
8112
Revised comment forms
fill out
Route
Invoke Comments form and route to users 1 and 2
8114
completed comments form
8118
Comments form
8116
Route
route
Store
8120
User 2 (experts)
fill out
Route
Route
Revised comment form
completed comments form
8124
8122
Store
Store
Store
Comments Form Database
8104

FIG. 81

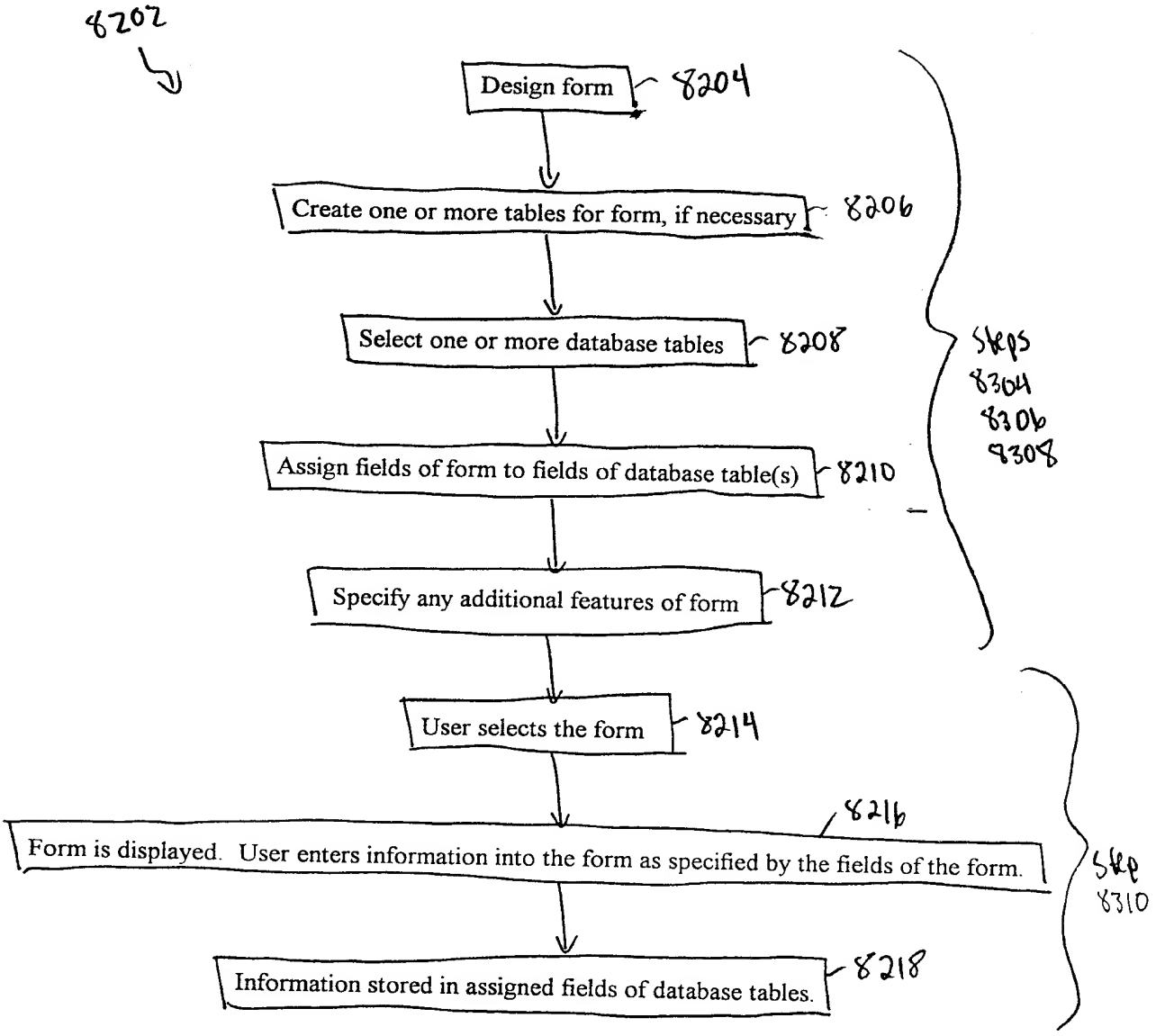

8202
Design form
8204
Create one or more tables for form, if necessary
8206
Select one or more database tables
8208
Assign fields of form to fields of database table(s)
8210
Specify any additional features of form
8212
Steps 8304 8306 8308
User selects the form
8214
8216
Form is displayed. User enters information into the form as specified by the fields of the form.
Step 8310
Information stored in assigned fields of database tables.
8218

FIG. 82

8302

8304

The system administrator authorizes a given user to create forms. In other words, the ability to create, modify, edit, and otherwise manage forms is preferably a secured operation. This is because the creation of a form will include appropriate provisions for storing the data acquired through the use of a form in a relational database for back-end statistical processing.

8306

The form-creator logs into the system and creates a form to capture the data they are interested in capturing. The invention supports several different types of form input widgets, including radio buttons (choose one from many items), text input fields, check boxes (represents "Yes/No" or "True/False") and so forth. The form-creator explicitly or implicitly associates input widgets with the back-end database tables stored on the server. Even the meta-content of the tables may be stored with the form, allowing for the use of sophisticated data mining applications of the form and the data. It is also possible to associate programmatic operations with the form, thereby allowing the form-creator to control what the annotation-creator sees while the annotation-creator is using the form. Examples of such operations would include "edit checks" on input fields (e.g., the form-creator could specify that an annotation-creator must enter in a number between 0 and 100 when entering data into the form).

A

FIG. 83A

64/99

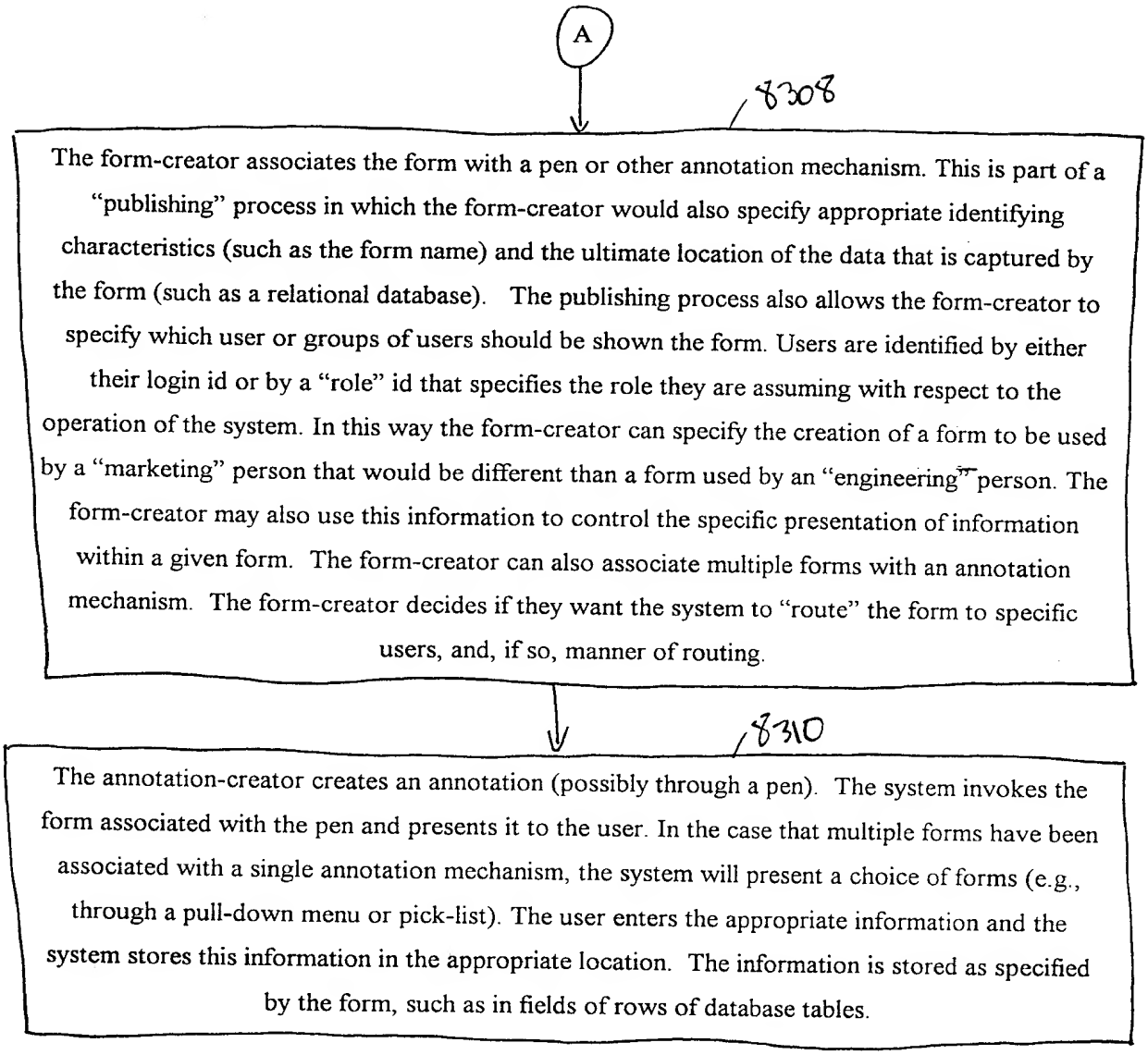

A
8308
The form-creator associates the form with a pen or other annotation mechanism. This is part of a "publishing" process in which the form-creator would also specify appropriate identifying characteristics (such as the form name) and the ultimate location of the data that is captured by the form (such as a relational database). The publishing process also allows the form-creator to specify which user or groups of users should be shown the form. Users are identified by either their login id or by a "role" id that specifies the role they are assuming with respect to the operation of the system. In this way the form-creator can specify the creation of a form to be used by a "marketing" person that would be different than a form used by an "engineering" person. The form-creator may also use this information to control the specific presentation of information within a given form. The form-creator can also associate multiple forms with an annotation mechanism. The form-creator decides if they want the system to "route" the form to specific users, and, if so, manner of routing.
8310
The annotation-creator creates an annotation (possibly through a pen). The system invokes the form associated with the pen and presents it to the user. In the case that multiple forms have been associated with a single annotation mechanism, the system will present a choice of forms (e.g., through a pull-down menu or pick-list). The user enters the appropriate information and the system stores this information in the appropriate location. The information is stored as specified by the form, such as in fields of rows of database tables.

FIG. 83B

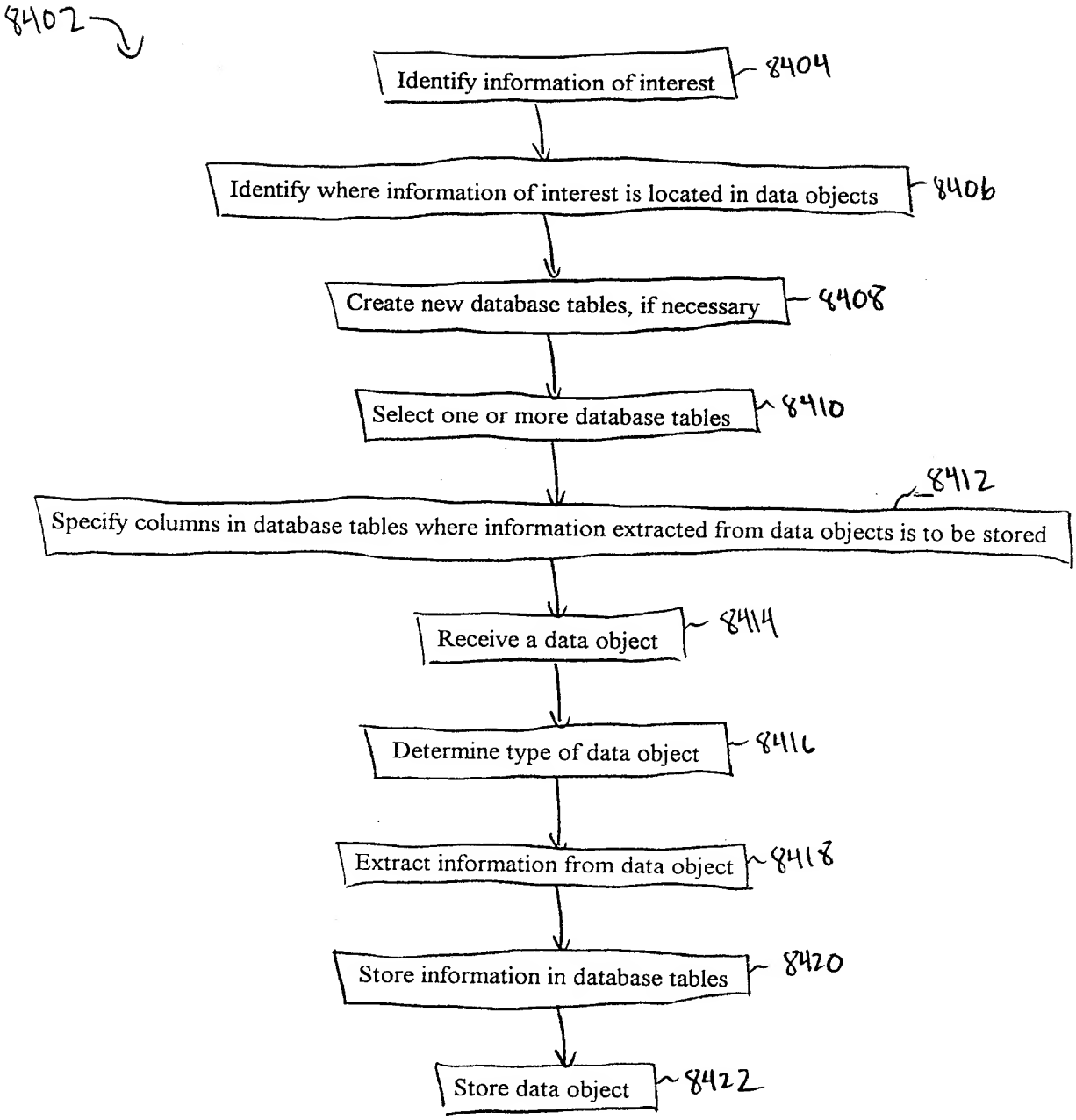

8402
Identify information of interest
8404
Identify where information of interest is located in data objects
8406
Create new database tables, if necessary
8408
Select one or more database tables
8410
Specify columns in database tables where information extracted from data objects is to be stored
8412
Receive a data object
8414
Determine type of data object
8416
Extract information from data object
8418
Store information in database tables
8420
Store data object
8422

FIG. 84

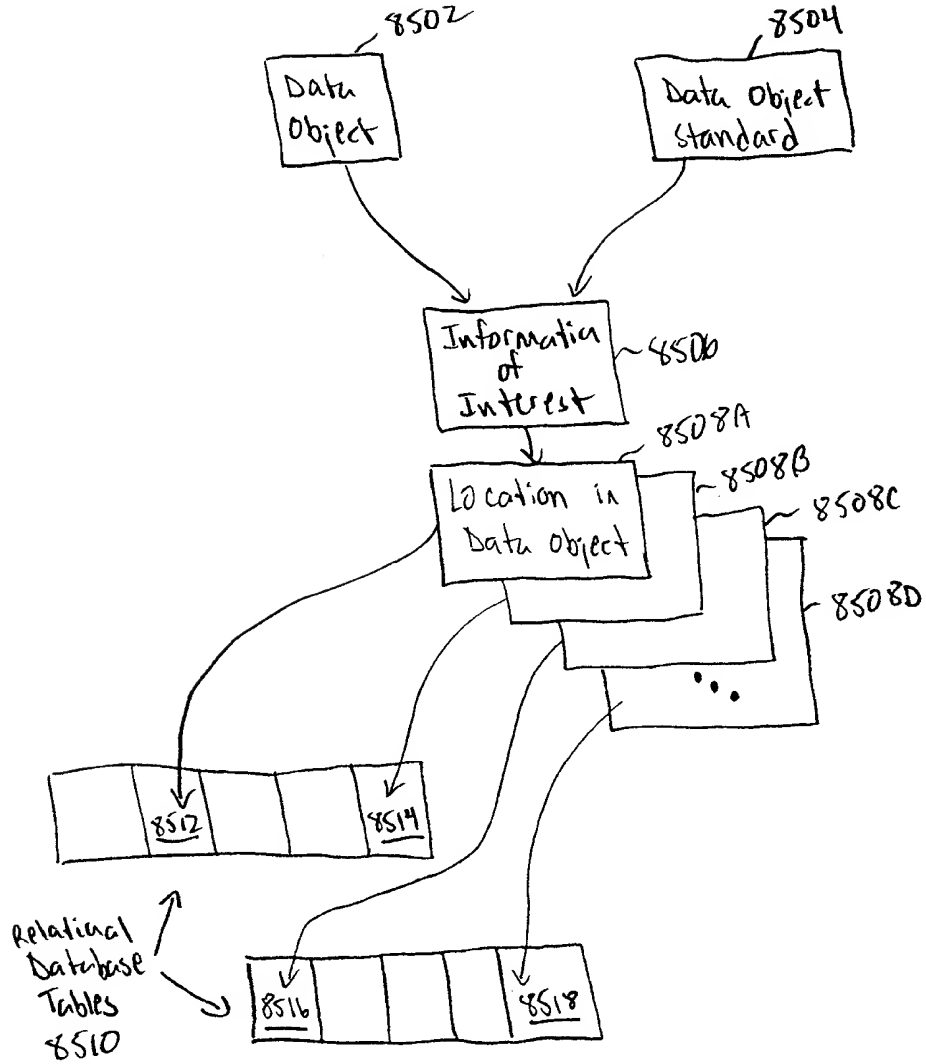

8502
Data Object
8504
Data Object Standard
Information of Interest
8506
Location in Data object
8508A
8508B
8508C
8508D
8512
8514
Relational Database Tables 8510
8516
8518

FIG. 85

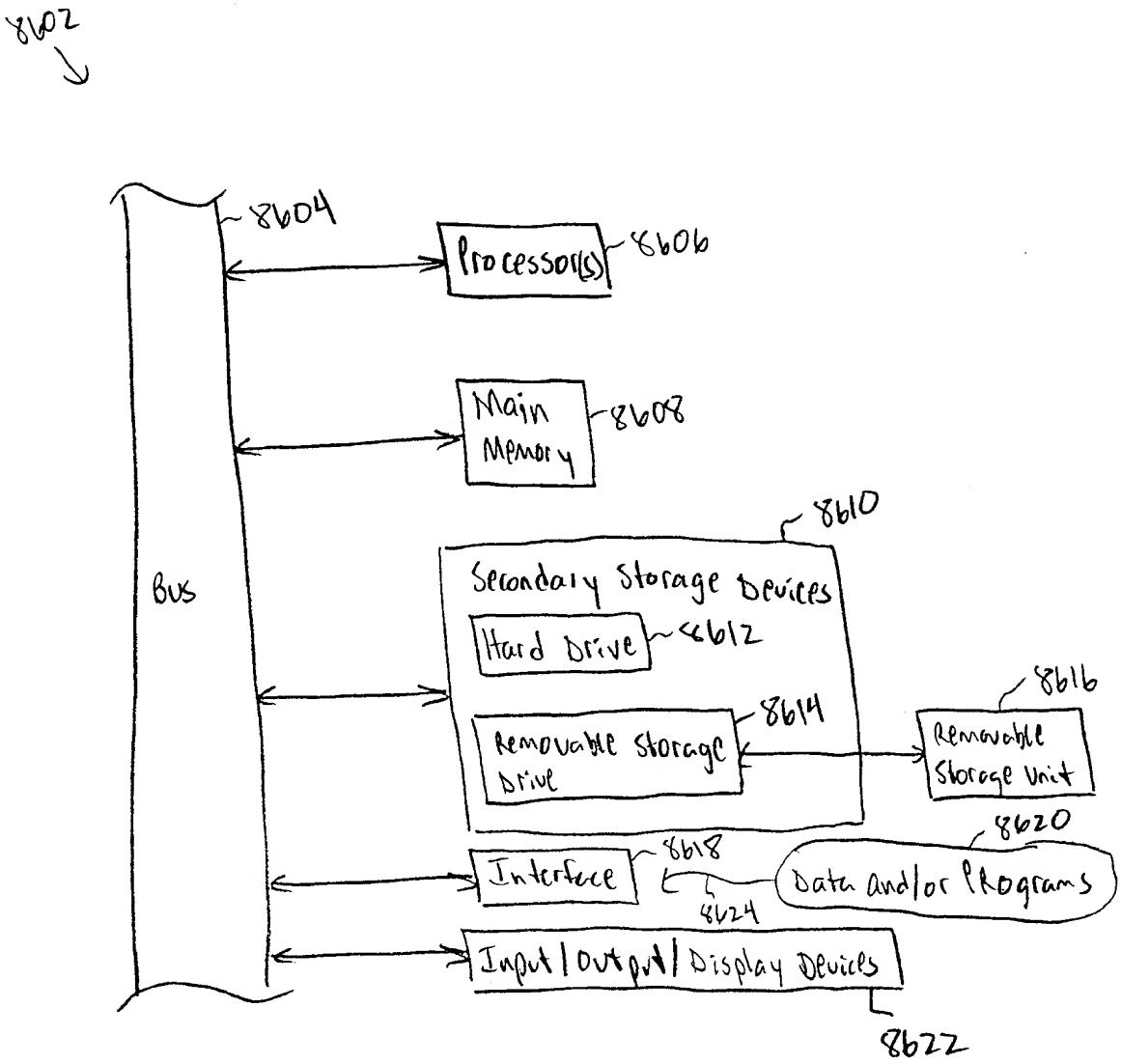

8602
8604
Processor(s)
8606
Main Memory
8608
8610
Secondary Storage Devices
Hard Drive
8612
Removable Storage Drive
8614
8616
Removable Storage Unit
Bus
Interface
8618
8620
Data and/or Programs
8624
Input/Output/Display Devices
8622

FIG. 86

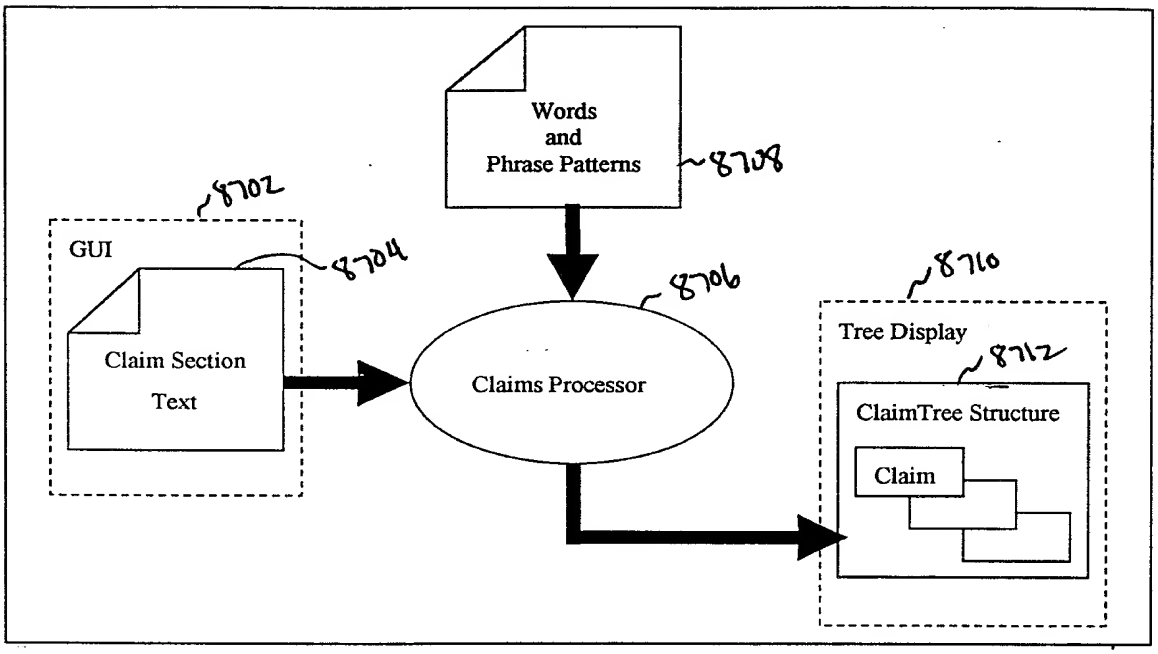

Words
and
Phrase Patterns
8708
8702
GUI
8704
Claim Section
Text
8706
Claims Processor
8710
Tree Display
8712
ClaimTree Structure
Claim

FIG. 87

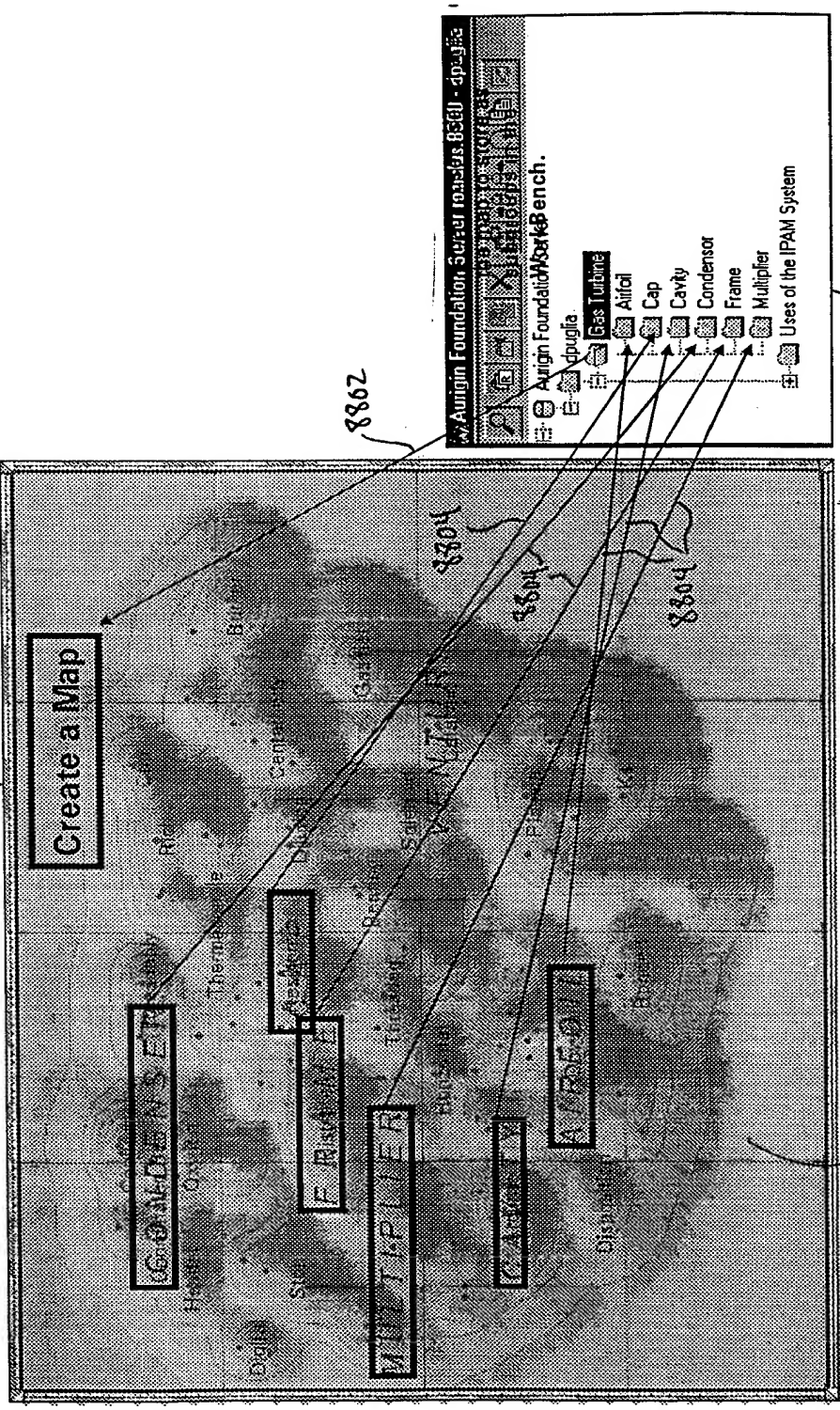

Create a Map
Aurigin Foundation Server master 8560 - dpuglia
Aurigin Foundation Workbench.
dpuglia
Gas Turbine
Airfoil
Cap
Cavity
Condensor
Frame
Multiplier
Uses of the IPAM System
8802
8804
8804
8804
Landscape Tool 808
IPAM 806
Map 810

FIG. 8

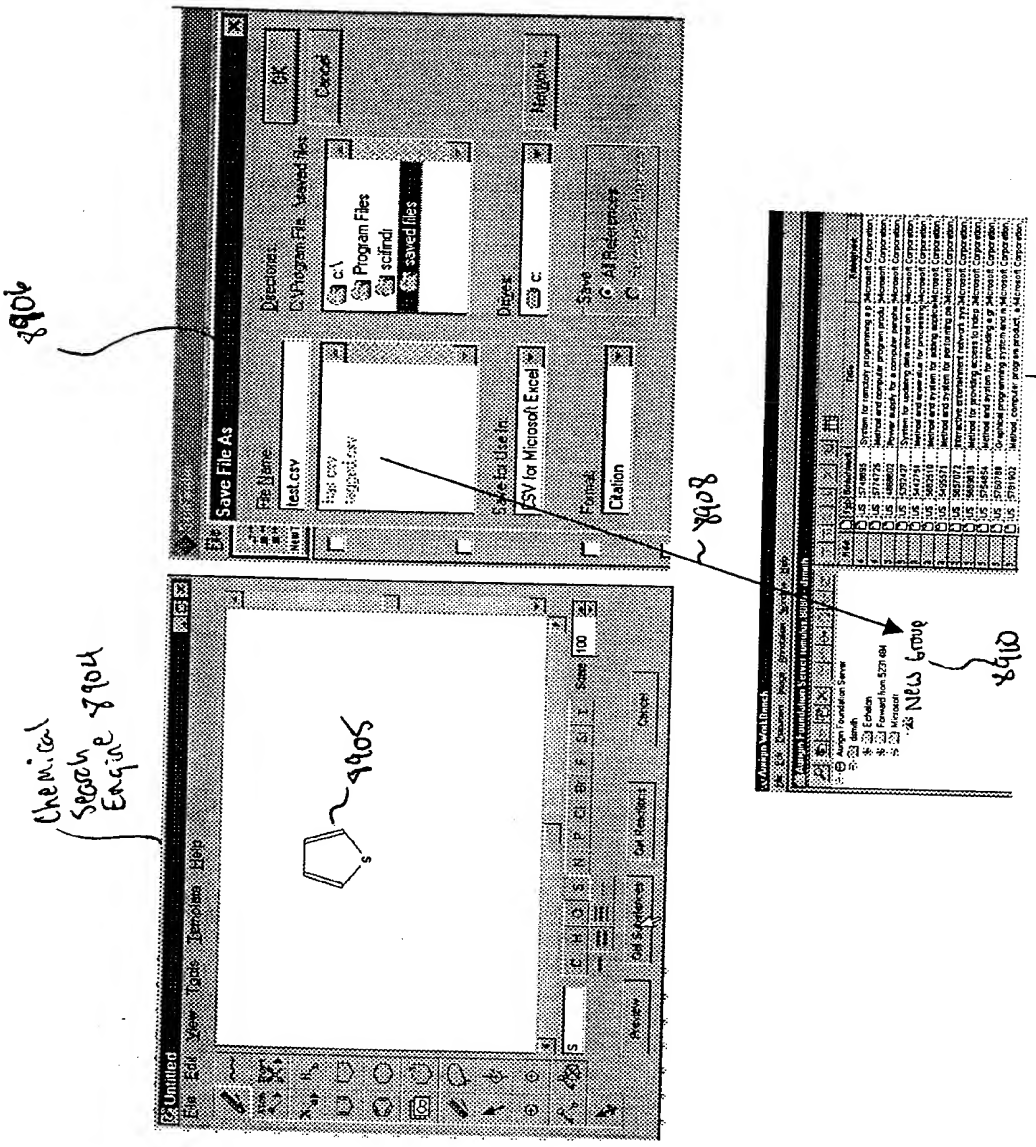

Chemical Search Engine 9904
9905
9906
Save File As
File Name:
test.csv
Directories:
C:\Program Files\...\saved files
Program Files
scfind
saved files
Drives:
c:
Save as type:
CSV for Microsoft Excel
Format:
Citation
OK
Cancel
Network...
9908
9910
News group
IDAM 9902

Fig. 9A

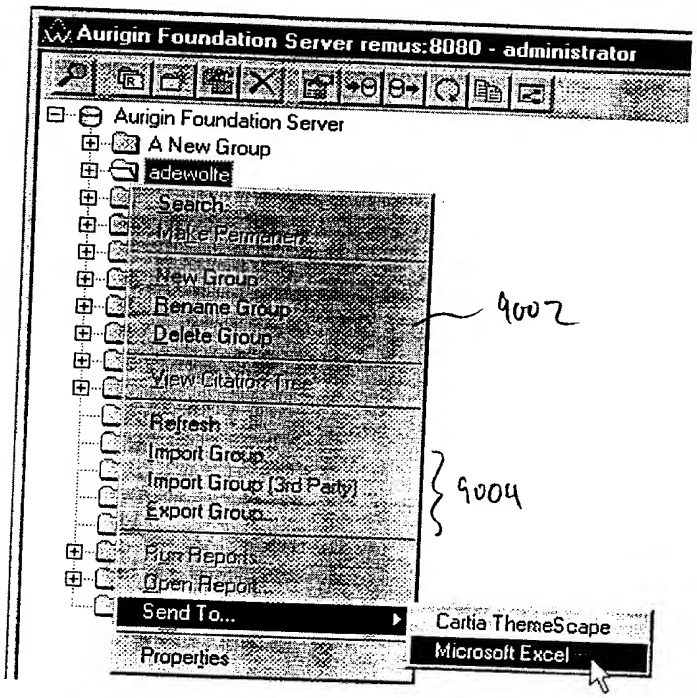

Aurigin Foundation Server remus:8080 - administrator
Aurigin Foundation Server
A New Group
adewolfe
Search...
Make Permanent
New Group...
Rename Group
Delete Group
View Citation Tree
Refresh
Import Group...
Import Group [3rd Party]
Export Group...
Run Reports...
Open Report...
Send To...
Properties
Cartia ThemeScape
Microsoft Excel
9002
9004

FIG. 9D

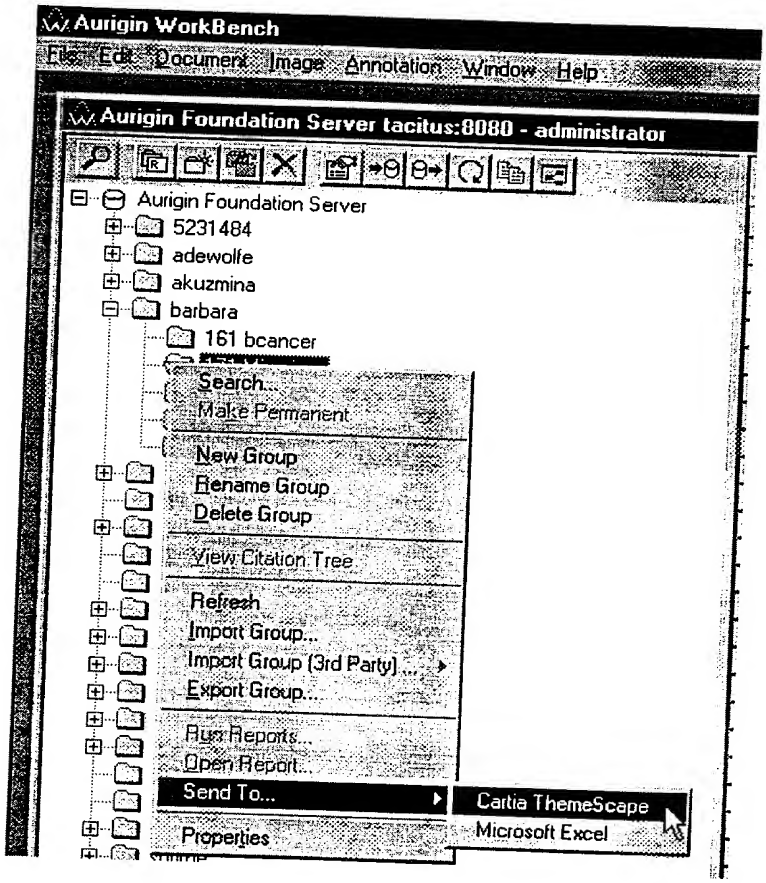

Aurigin WorkBench
File Edit Document Image Annotation Window Help
Aurigin Foundation Server tacitus:8080 - administrator
Aurigin Foundation Server
5231484
adewolfe
akuzmina
barbara
161 bcancer
Search...
Make Permanent
New Group
Rename Group
Delete Group
View Citation Tree
Refresh
Import Group...
Import Group (3rd Party)...
Export Group...
Run Reports...
Open Report...
Send To...
Properties
Catia ThemeScape
Microsoft Excel

FIG. 9I

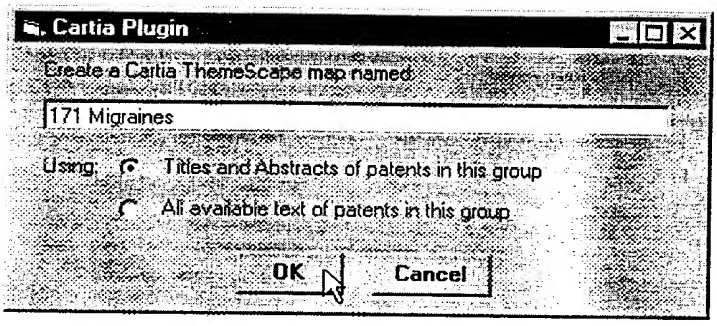

Cartia Plugin
Create a Cartia ThemeScape map named:
171 Migraines
Using:
Titles and Abstracts of patents in this group
All available text of patents in this group
OK
Cancel

FIG. 92

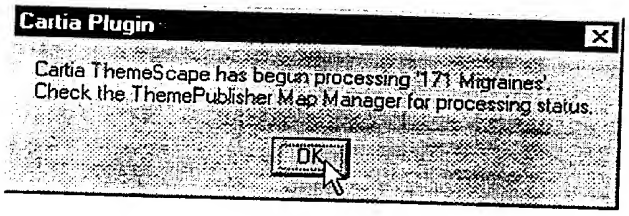

Cartia Plugin
Cartia ThemeScape has begun processing '171 Migraines'.
Check the ThemePublisher Map Manager for processing status.
OK

FIG. 93

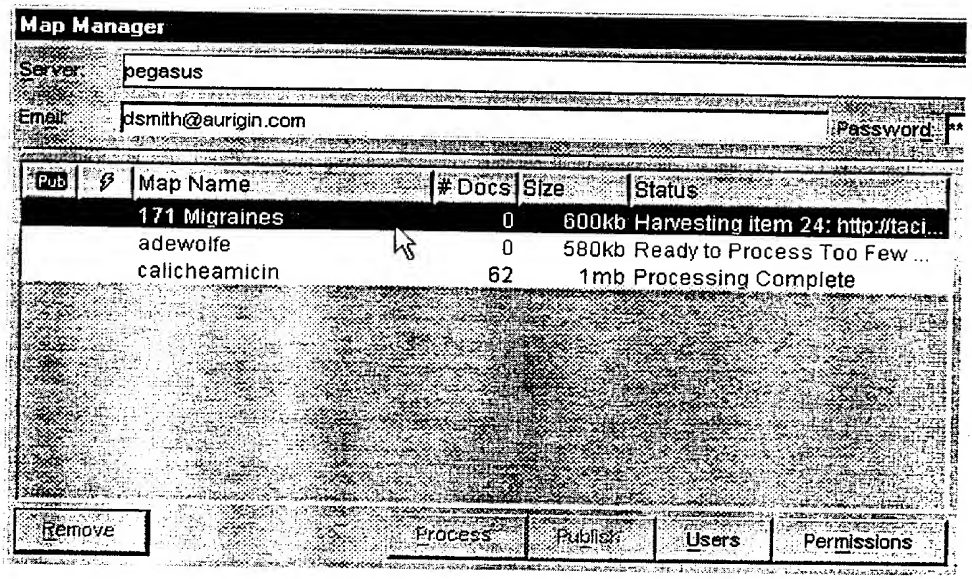

Map Manager
Server: pegasus
Email: dsmith@aurigin.com
Password: **
Pub
Map Name
Docs
Size
Status
171 Migraines 0 600kb Harvesting item 24: http://taci...
adewolfe 0 580kb Ready to Process Too Few ...
calicheamicin 62 1mb Processing Complete
Remove
Process
Publish
Users
Permissions

FIG. 94

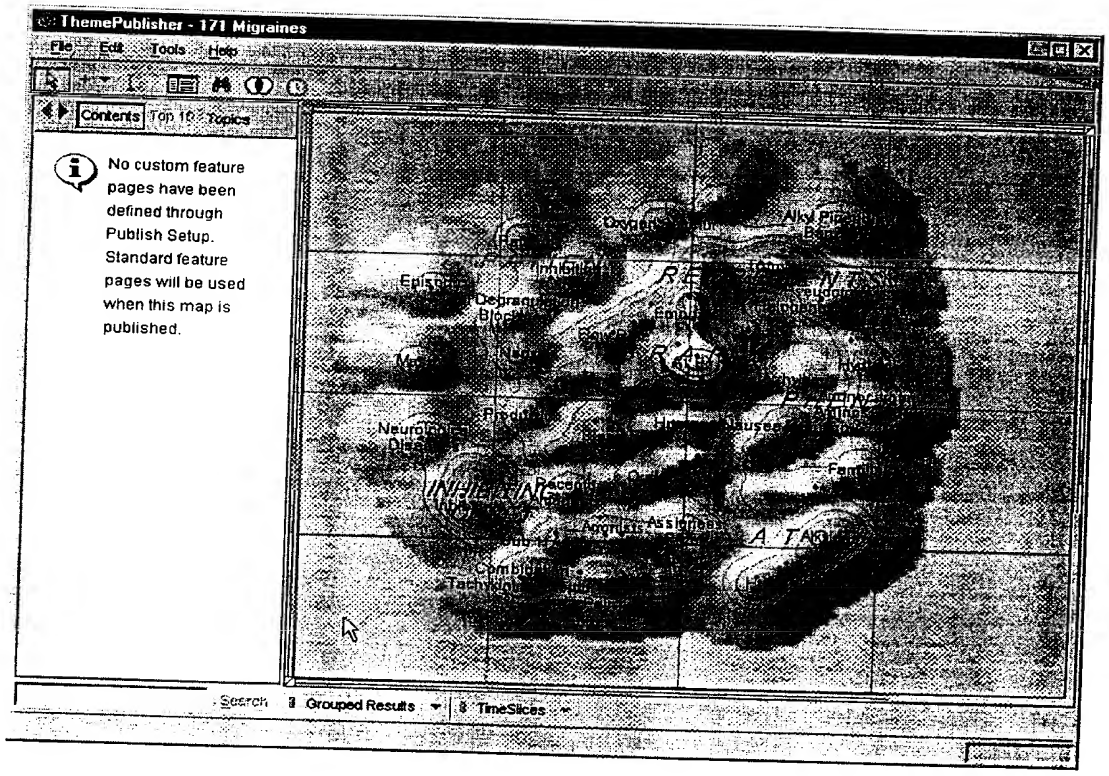

ThemePublisher - 171 Migraines
File Edit Tools Help
Contents Top 10 Topics
No custom feature pages have been defined through Publish Setup. Standard feature pages will be used when this map is published.
Search
Grouped Results
TimeSlices

FIG. 95

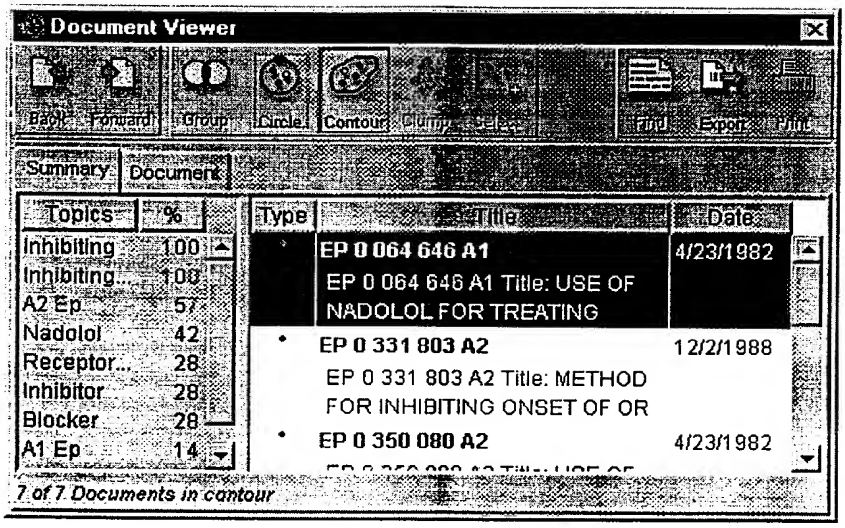

Document Viewer
Back
Forward
Group
Circle
Contour
Clump
Select
Find
Export
Print
Summary
Document
Topics
%
Inhibiting 100
Inhibiting... 100
A2 Ep 57
Nadolol 42
Receptor... 28
Inhibitor 28
Blocker 28
A1 Ep 14
Type
Title
Date
EP 0 064 646 A1
EP 0 064 646 A1 Title: USE OF NADOLOL FOR TREATING
4/23/1982
EP 0 331 803 A2
EP 0 331 803 A2 Title: METHOD FOR INHIBITING ONSET OF OR
12/2/1988
EP 0 350 080 A2
4/23/1982
7 of 7 Documents in contour

FIG. 9b

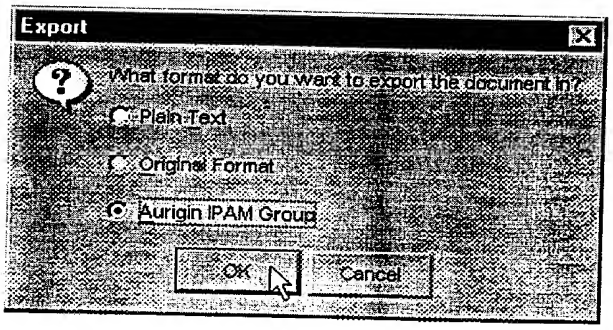

Export
What format do you want to export the document in?
Plain Text
Original Format
Aurigin IPAM Group
OK
Cancel

FIG. 97

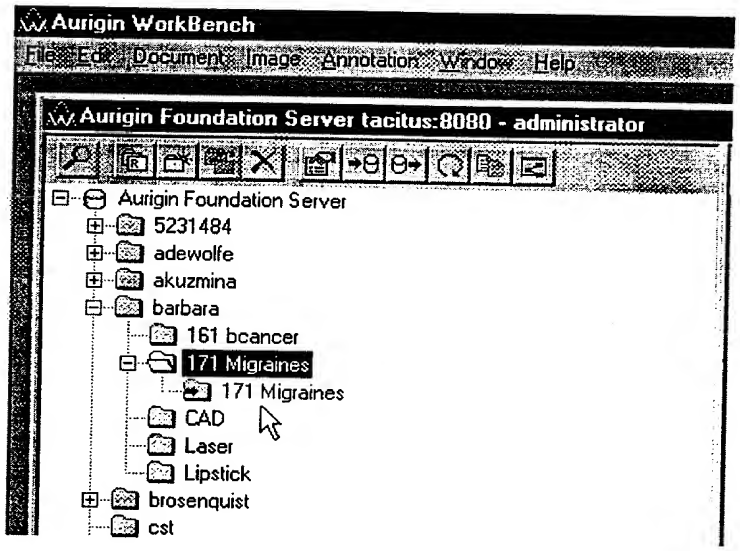

Aurigin WorkBench
File Edit Document Image Annotation Window Help
Aurigin Foundation Server tacitus:8080 - administrator
Aurigin Foundation Server
5231484
adewolfe
akuzmina
barbara
161 bcancer
171 Migraines
171 Migraines
CAD
Laser
Lipstick
brosenquist
cst

FIG. 9B